Protein Structure Prediction Using Honey-Bee Mating Optimization

A. Boukra[1], S. Bouroubi[2]
[1]USTHB, Faculty of Electronics and Computer Science
Laboratory LSI
USTHB, Faculty of Mathematics
[2]Department of Operational Research
Laboratory LAID3
[1,2]BP 32 16111 El-Alia, Bab-Ezzouar
Algiers, Algeria
{amboukra, bouroubis}@yahoo.fr

ABSTRACT: *Protein folding prediction is a fundamental problem in biology. The spatial structure (Conformation) of the protein is the manipulation key of its biochemical and cellular functions. The protein tends to fold on itself. However, a protein cannot fold in any way: it must reach a lowest level of energy. Determining the conformation of the protein in this state of lowest energy is known as the protein folding problem. This can be modeled as an optimization problem. Even under the simplified models, the problem is NP-complete [1] [2] [3]. Thus, there is no polynomial time algorithm to resolve this problem. In this paper, a biologically inspired algorithm for protein spatial structure prediction is proposed; it uses the honey-bee colony reproduction processes.*

Keywords: Protein Folding, Conformation, Honey-Bee Mating Optimization, HP model, NP-Complete

Received: 2 June 2012, Revised 2 July 2012, Accepted 14 July 2012

1. Introduction

The study and resolution of the protein folding problem is of interest in various fields such as human health and biotechnology. In the field of human health, the ability to solve this problem could contribute to the prediction and treatment of diseases. In biotechnology, it could allow the realization of large-scale enzymatic processes and the development of protein drugs [4] [5] [6]. Protein folding is the physical process by which a polypeptide (straight chain of amino acids) folds into a three dimensional structure, in which it is functional. Correct three-dimensional structure, or native structure, is essential for performing functions within the cell. A protein can adopt different conformations. Like a stretched spring, a protein in unstable state seeks to achieve its most stable state which is characterized by minimum energy. The native structure of a protein is generally at its lowest energy conformation. It is established that the amino acid sequence is the major factor in determining the conformation of a protein [7]. The experimental determination of the three-dimensional structure of a protein is often very difficult and costly. Using algorithms to predict the three dimensional structure of the protein represents a good alternative. The number of possible conformations for chains of amino acids is very large; the use of metaheuristics is required. Several studies using metaheuristics have been proposed. Among these we find the use of genetic algorithms in [8], [9], [10] and [11]. Memetic algorithms have been used in [12]

and [13]. Immune algorithms have been used in [14] [15]. In [16] a randomized multiStart tabu search algorithm has been applied. A hybrid approach combining genetic algorithms and tabu search has been proposed in [17]. ACO has been used in [18] [19] [20] and PSO has been used in [26]. The rest of the paper is organized as follows. In Section 2, the protein folding problem is modeled. The representation of the solution is presented in Section 3, section 4 present the calculation of the fitness, followed by the presentation in Section 5 of the MBO algorithm. Section 6 presents the experimental study and its results. We finally conclude in Section 7.

2. Modeling the Protein Folding Problem

Modeling the protein folding problem starts with a model for the protein itself and a model for the conformational space**.**

2.1 Protein Modeling

Modeling a protein can be a full representation of amino acids at the atomic level. Obviously, such a model would result in a significant computational cost. The HP model [21] represents a good compromise between the modeling simplicity and the non degradation of the quality of the results. The HP model [21] [22] suggests reducing the amino acids at only two groups: hydrophilic amino acids (called P) and hydrophobic amino acids (called H). Thus, a protein, rather than being represented by a sequence of several different amino acids, will simply be represented by a sequence of amino acids H or P. In addition, an amino acid is reduced to a single point in space; it does absolutely not consider the atomic detail. This model allows us to model this problem as a combinatorial optimization problem.

The model has the following characteristics:

• Amino acids (monomers) all have the same size.

• Amino acids are simplified. There are only 2 types: Hydrophobic monomers (H) and Hydrophilic monomers (P)

• Links between two monomers have the same length.

• Each monomer can occupy a position on a two- or three-dimensional grid.

• The function calculating the energy of a protein in a given conformation is described as follows:

Let a and b be two nonconsecutive H monomers in the protein and f a function such that, f (a, b) = – 1 if a and b are neighbors in the conformation, and f (a, b) = 0 otherwise. The energy of the protein in this conformation is equal to the sum of f (a, b) on all nonconsecutive pairs a and b. The problem is therefore to find the lowest energy conformation, which would maximize the number of HH contacts [21] (Figure 1).

(a) Number of HH contacts = 0 (b) Number of HH contacts = 1

Figure 1. Example showing two conformations of the same chain with a different number of HH contacts

2.2 Modeling the Conformations Space

For modeling the conformational space, there are basically two models [23]:

2.2.1 Network model

In this model, we use a (2D or 3D) grid, where the distances between the monomers are identical and the angles between them are 90°.

2.2.2 Off-grid models

The particular feature in this model is that the angles between the monomers are arbitrary and that the distances are not identical. Calculations in this model are time-consuming.

In what follows we have opted to retain the HP model for the protein and the network model for the conformations space.

3. Representation of the Solution

There are two main ways of representing the conformation of a protein. The first is to use absolute coordinates, i.e. the abscissa and ordinate of each monomer of the protein sequence. The second uses relative coordinates, that is to say the position of a monomer compared to the previous two monomers. In this case, three possible directions can be envisaged: in the same direction as the previous two monomers (coded 0), 90°clockwise (coded 1), or 90° counterclockwise (coded 2).We have used the two representations in our approach.

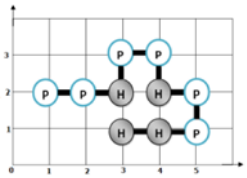

Figure 2. Representation of the solution in absolute coordinates

The solution in absolute coordinates for Figure 2 is represented as follows: (1, 2), (2, 2), (3, 2), (3, 3), (4, 3), (4, 2), (5, 2), (5, 1), (4, 1), (3, 1) .

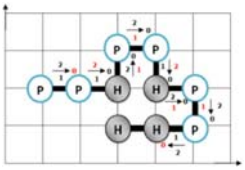

Figure 3. Representation of the solution in relative coordinates

The solution in relative coordinates for Figure 3 is represented as follows: 0, 2, 1, 1, 2, 1, 1, 0.

We have used the representation in absolute coordinates to check the feasibility and calculate the cost of conformation. We have used the representation in relative coordinates to represent the solution manipulated in MBO.

4. Calculating the fitness of a solution

Computing the fitness of a solution is based on the location of the H monomers. Fitness is the number of H monomers, non-consecutive in the chain, that are adjacent in the grid. The calculation procedure is based on the table of absolute coordinates. The adjacency of two H monomers is summarized in Table 1.

difference between the abscissas	difference between the ordinates	adjacency
1	0	yes
0	1	yes
0	0	no
1	1	no
Other values	Other values	No

Table 1. Adjacency of two H monomers

5. Presentation of the MBO Algorithm

MBO [24] is a metaheuristic inspired by the process of bee reproduction and evolution. Compared to other evolutionary methods, the main characteristic of MBO is the possibility of combining the principle of evolutionary metaheuristics and other heuristics. These heuristics represent the workers who are used to improve solutions. The algorithm is given bellow.

Algorithm 1: MBO algorithm

```
Begin
    Initialize the parameters
    Randomly generate the queens
    For a pre-defined maximum number of mating-flights
        For each queen in the queen list
            Initialize energy, speed and step
            Generate a drone
            While energy > 0
                Probabilistically chooses drones
                If a drone is selected
                    Then add its sperm to the queen's spermatheca
                End if
                Update the queen's internal energy and speed
                Mutate the drone
            End While
        End for
        Generate broods by crossover and mutation
        Use workers to improve the broods.
        Replace the least-fit queen with the best brood
        Remove the best brood from the brood list
    End for
End.
```

Our goal, in this work, is to propose an adaptation of this algorithm to the problem. The main difference between the original algorithm and our algorithm is the use of the workers after generating each queen and each egg. This avoids the problem of discontinuity in the conformation and improves the quality of the solutions. Each phase of the algorithm is described below.

5.1 Initialization

The first phase of the algorithm is to initialize the six parameters: number of queens, number of flights by queen, number of broods per queen, number of workers (number of heuristics used), number of brood improvements (by workers) and the size of the spermatheca.

5.2 Generating a random set of queens

This set represents the initial feasible solutions of the problem. We use the following algorithm to randomly generate feasible queens.

Algorithm 2: Queen generating algorithm

```
Begin
    Put the first two monomers on the grid
    Update the table of absolute coordinates
    While (not end of the chain)
        Select the next monomer
        Generate randomly a direction belonging to {0, 1, 2} witch position in not occupied on the grid
        If such position exist then
                Put the monomer on the grid
                Update the table of absolute coordinates
            Else generate another queen
        End if
    End while
End.
```

5.3 Initialization of energy, speed, and step of the Queen

At the beginning of each mating flight, the values of energy and speed of the queen are initialized randomly in the interval [0.5, 1]. Step is the amount of energy expanded after each coupling; it is given by Formula (1).

$$Step = \frac{0.5 \times Initial\ energy}{Spermatheca\ Size} \quad (1)$$

5.4 Generating a Drone

A drone D consists of a genotype and a mask used to hide half of its genes. The generation of the genotype and the mask is performed randomly.

5.5 Selecting the drone for the mating

A drone is selected for mating according to the probability prob (Q, D) given in (2).

$$prob\,(Q, D) = e^{-\frac{Difference}{Speed}} \quad (2)$$

Where *difference* is the absolute difference between the fitness of the genotype of queen Q and drone D. The genotype of the selected drone is placed in the queen spermatheca.

5.6 Update of energy and speed

After each mating between a queen and a drone, the speed and the energy of the queen are diminished by Formulas (3) and (4), respectively:

$$Speed\,(t+1) = \alpha \times Speed\,(t) \quad (3)$$

α : speed reduction factor between 0 and 1

$$Energy\,(t+1) = Energy\,(t) - Step \quad (4)$$

5.7 Drone genotype mutation

The mutation mechanism is to change certain genes of the genotype with a probability that uses the speed of the queen. We generate a random number between 0 and 1. If this number is less than the speed of the queen, the mutation takes place. The purpose of this mutation is to generate a new drone capable of mating with the queen.

We repeat the execution of the algorithm from step **e** until the energy of the queen becomes zero.

5.8 Crossover

At the end of step **g**, each queen performs a crossover between her genotype and those present in her spermatheca. Each crossover gives an egg (solution represented by a genotype). The number of genotype in the spermatheca may be reduced. The same genotype can therefore be selected several times for the crossing. To avoid having the same offspring, we opted for a crossover at random points.

5.9 Improvement

The genotypes resulting from the previous step are improved by the workers (problem specific heuristics). For our problem, the role of workers is to correct and improve the eggs (i.e. the solutions resulting from the crossing of the queen with the genotypes of drones). The correction approach that we have adopted is to keep the correct parts of the chain and correct the parts containing monomers overlays. It should be noted that taking into account all possible corrections is expensive. For this reason, we have opted for a fixed number of corrections (a parameter giving the number of improvements). The best correction will be retained. Our approach is to make a series of random changes of bits that caused the infeasibility. Each change must verify the feasibility of the solution.

5.10 Replacement

In this step, the bad queens of the current population are replaced by the best broods found in step **i**.

The algorithm ends when all the mating flights are completed.

6. Experimental Study

The experiment has been performed on a computer with an Intel Pentium 4 processor clocked at 3.0 GHz with 1 GB memory. The

programming has been done with the Java language.

To validate our approach, a comparison with benchmarks is needed. The chains of proteins taken into account by the benchmarks are limited in size (≤ 100) and have specific configurations (Table 2) [25]. Table 2 is composed of 04 columns. The first column represents the number of the protein and the second column contains the structure of the protein. The index used in the protein structure represents the repetition factor of the sequence, $(HP)_2$ is equivalent to HPHP. The fourth column represents the length of the protein, and the last column represents the optimal number of H-H contacts, found in [25] (this value corresponds to minimum energy conformation).

N°seq	protein	L	NB_contacts H-H
1	$(HP)_2\,PH\,(HP)_2\,(PH)_2\,HP\,(PH)_2$	20	9
2	$H\,(HP_2)_7\,H_2$	24	9
3	$(P_2H)_2\,H\,(P_4H_2)_3$	25	8
4	$P\,(P_2H_2)_2\,P_5\,H_7\,P_2\,H_2\,P_4\,H\,(HP_2)_2$	36	14
5	$P_2\,H\,(P_2H_2)_2\,P_5\,H_{10}\,P_6\,(H_2P_2)_2\,H\,P_2\,H_5$	48	23
6	$H_2(PH)_4\,H_3\,P\,H(P_3H)_2\,P_4\,(HP_3)_2\,HP\,H_4\,(PH)_4\,H$	50	21
7	$P_2H_3PH_8P_3H_{10}PHP_3H_{12}P_4H_6PH_2PHP$	60	36
8	$H_{12}(PH)_2((P_2H_2)_2P_2H)_3(PH)_2H_{11}$	64	42
9	$H_4P_4H_{12}P_6(H_{12}P_3)_3HP_2\,(H_2P_2)_2HPH$	85	53
10	$P_3H_2P_2H_4P_2H_3(PH_2)_3H_2P_8H_6P_2H_6P_9HPH_2PH_{11}P_2H_3PH_2PHP_2HPH_3P_6H_3$	100	50

Table 2. Benchmarks used [25]

Before comparing the MBO results with benchmarks, we adjust the MBO parameters.

6.1 MBO Parameters Setting

Adjusting the parameters of a metaheuristic is done experimentally or through another metaheuristic that can be the metaheuristic itself (case of adaptive and self adaptive meta heuristics). We have used the Hill Climbing algorithm to adjust MBO parameters. This algorithm improves an existing solution, by searching in its neighborhood. The stagnation in a local optimum is the major disadvantage of this algorithm. We have noted that the increase of the parameter values has the effect of improving the solution until a stagnation state. Table 3 summarizes the parameters found for each chain of the Benchmark.

N° of chain	Nb-Queen	Nb-fligh	spermatheca -size	Nb-Brood	Nb-improvement	Nb-failure
1	30	20	100	40	40	446
2	30	20	100	60	80	540
3	30	20	100	80	80	267
4	50	20	100	150	250	698
5	50	20	200	500	500	780
6	50	20	200	500	500	445
7	50	20	200	600	600	456
8	50	20	200	600	600	686
9	50	20	200	600	600	522
10	50	20	200	600	600	728

Table 3. MBO Parameter setting

The meaning of each parameter is given below.

• **NB-queen** represents the number of queens. This parameter determines the number of algorithm iterations. Number of iterations = (NB-queens × NB-flights).

• **NB-flights** represents the number of flights for each queen.

• **Spermatheca-size** limits the number of drones in the queen spermatheca and determines the step (amount of energy to decrease).

• **NB-broods** represents the size of the final population.

• **NB-improvement** represents the number of iterations to be done to correct and improve a brood.

• **NB-failures** is used to stop the process if there is no improvement of the solution after a number of iterations.

For good diversification, at the beginning of the algorithm, the probability of accepting solutions must be high. The probability (2) gives a high value if the speed is maximum. We have fixed the value of ± to 0.9.

We used a single worker. Its role is to ensure the feasibility of the conformation and improve the solutions.
To validate our approach for larger sizes and random configurations, we have introduced the notion of upper bound (M).

6.2 Calculation of the Upper Bound

To calculate the upper bound M, we have first noted the following: the distance between two H monomers that may be adjacent is always odd and the difference is equal to 1. A monomer which is located at an even distance from another monomer can therefore never be adjacent (See Figures 4 and 5).

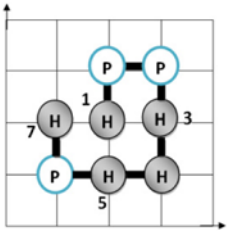

Figure 4. Adjacency of H monomers (odd distance)

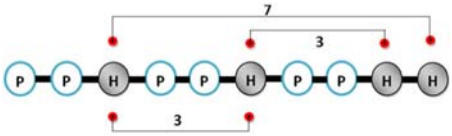

Figure 5. Example of adjacency of H monomers (odd distance in the chain)

6.2.1 Procedure for calculating the bound M

1. Number the H monomers. (Figure 6).

2. Consider that each H is a vertex of a graph G.

3. Each vertex not corresponding to the first or the last H of the chain can be linked to two edges. The vertices corresponding to the first or the last H of the chain are linked to at most three edges.

4. Draw an edge between two vertices of G if the distance between the two vertices corresponding to monomers in the chain is odd (Figure 7).

Execute step (4) beginning with the first summit after the second, etc., without violating Condition (3). The number of edges in the constructed graph is the upper bound M for the chain.

Example:

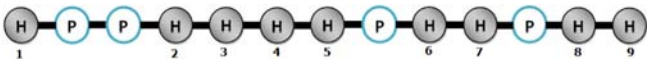

Figure 6. Numbering H monomers in the chain

This bound can be used to evaluate the quality of the solution found by MBO. Let X be the solution found by MBO, X* the optimal solution, M the upper bound, and f (X) the fitness of a solution. It is clear that $f(X) \le f(X^*) \le M$ and $R = \frac{f(X)}{M} \le 1$. If R

is close to 1, then f (X) is close to M, and since f (X) ≤ f (X *) ≤ M then f (X) is close to f (X *). However, if R is not close to 1, we cannot conclude anything. For protein chains longer than 100, we calculate R = $\frac{f(X)}{M}$.

6.3 Comparison of MBO Results with Benchmark Results

We compare the results of MBO with the optimal solutions found in [25].

N°seq	protein size	optimal solution	MBO solution	bound M	Ratio MBO/opt	Ratio opt/M
1	20	9	9	9	100%	100%
2	24	9	9	9	100%	100%
3	25	8	8	8	100%	100%
4	36	14	14	16	100%	87%
5	48	23	23	24	100%	96%
6	50	21	18	24	86%	87%
7	60	36	31	40	86%	90%
8	64	42	34	43	81%	98%
9	85	53	41	57	77%	93%
10	100	50	39	56	78%	89%

Table 4. Comparison of the MBO results with those of the benchmark

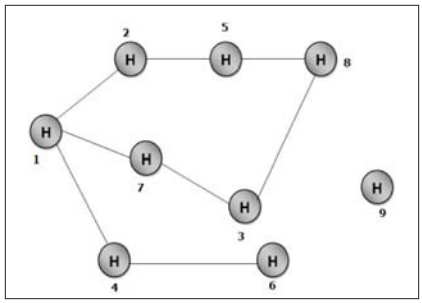

Figure 7. Construction of graph for calculating the bound (bound M = 8)

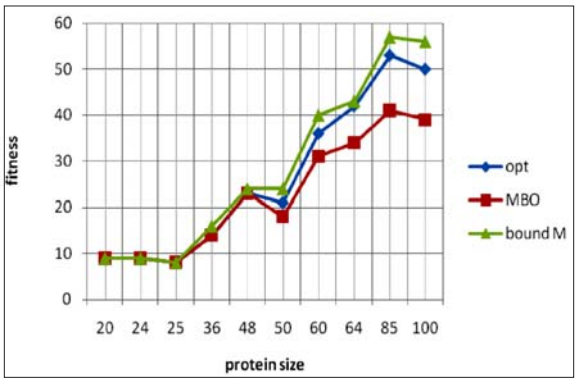

Figure 8. Comparative Graph between the MBO solutions, the benchmark and the bound

We note from Table 4 and Figure 8 that the optimal solution is reached for a chain length less than 48. We also note that the bound is equal to the optimum value for a chai n length less than 25.To better appreciate the quality of the MBO solutions, we calculate the performance ratio:

$$Performance\ ratio\ \frac{MBO}{OPT} = \sum_{i=1}^{10} \frac{ratio_i}{10} = 0.90$$

$$Performance\ ratio\ \frac{OPT}{M} = \sum_{i=1}^{10} \frac{ratio_i}{10} = 0.94$$

The ratios are high because we have used a small problem size. Therefore it indicates that MBO behaves well compared to the benchmark.

6.4 Comparison of MBO Results with Other Metaheuristics

We compare the MBO results to the r esults of ACO (Ant Colony Optimization), constraint programming (CP), the (hybrid) ant colony / constraint programming presented in [25]

chain	protein size	CP	ACO	CP/ACO	MBO	OPT
1	20	9	9	9	9	9
2	24	9	9	9	9	9
3	25	8	8	8	8	8
4	36	14	14	14	14	14
5	48	16	16	17	23	23
6	50	15	14	15	18	21
7	60	30	31	30	31	36
8	64	34	25	39	34	42
9	85	37	37	38	41	53
10	100	33	30	36	39	50

Table 5. Comparison between MBO, ACO, CP, ACO / CP [25]

We note (Table 5) that all the approaches s have found the optimum if the size of the chain is less than 36 (sequence 4). MBO found the optimum up to a size of 48 (sequence e 5). For chains of length greater than 36, MBO offers a better solution than the three other approaches, except for the sequence 8 w here CP / ACO provides a good result.

6.5 Experimentation of MBO for rand only generated chains

We have experimented MBO for randomly generated chains longer than100. For this we have used the Bound M. The results found are summarized in Table 6.

Protein Size	*MBO cost*	*bound*	*R=MBO/bound*
149	65	100	0.65
262	118	192	0.62
300	124	203	0.61

Table 6. Test for chains of length greater than 100

Beyond the length 300, the ratio R becomes small. In these conditions we cannot conclude anyt hing about the quality of the solution. For a chain of length less than 300, the ratio R is between 0.61 and 0.65. This means that the solutions found are quite close to the optimal solution.

6.6 Scalability Study

Table 7 and Figure 9 show the execution time of the algorithm for different problem sizes:

Chain size	Execution time
20	0.5
24	1.92
25	4.1
36	66
48	507
50	561
60	1351
64	1492
85	2021
100	3258

Table 7. Execution time (in min) for different problem sizes

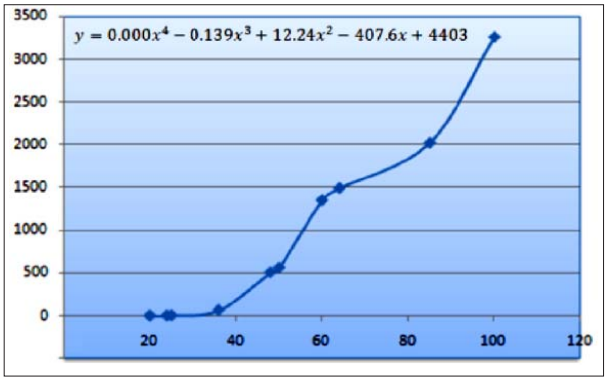

Figure 9. Representation of the execution time depending on the size of protein

We have approximated the equation of t he curve representing the execution time depending on the problem size. We have used the Microsoft Excel approximation tool. The equation found is: $y=0.0001x^4-0.139x^3+12.24x^2-40\ 7.6x+440$. The degree of the polynomial is equal to 4.

7. Conclusion

In this paper, we are interested in the protein folding prediction. We have used the HP two-dimensional model for the protein. The protein folding prediction being NP-complete, we have solved it using a metaheuristic inspired by the process of reproduction and evolution of bees, named MBO. The main contributions of this paper are as follows:

We have adapted MBO to the proteins folding problem, with a slight modification. This modification consists in using the workers at the time of generation of queens and the production of eggs. This avoids the problem of discontinuity in the conformation and improves the quality of the solutions. We have used the Hill Climbing algorithm to automatically adjust the MBO parameters for each chain. This avoids using the same parameters for all chains. We have shown that our results are very close to the exact results (performance ratio= 0.90) for small problems sizes. We have defined an upper bound for the value of the objective function and have used this upper bound to validate the results found by our approach for large instances. We have shown that for chains of length ranging from 149 to 300, the report performance varies between 0.61 and 0.65. This means that the solutions found by our approach for large problems sizes are quite close to the optimal solution. We have shown that MBO generally gives better results than those provided by CP, ACO and ACO / CP. Finally, we have shown that our approach is scalable. The equation representing the execution time of the algorithm with respect to the size of the problem is a polynomial of degree 4. Thus we believe that BBO represents a good approach for protein folding prediction.

Bibliography

[1] Berger B., Leighton, T. (1998).Protein folding in the hydrophobic-hydrophilic (HP) model is NP- complete, *Mathematics Department and Laboratory for Computer Science*, MIT, Cambridge, MA 02139, and USA.PMID: 9541869

[2] Alena Shmygelska, (2005). Holger H Hoos An ant colony optimization algorithm for the 2D and 3D hydrophobic polar protein folding problem. BMC Bioinformatics; doi: 10.1186/1471-2105-6-30, (February 2005). Department of Computer Science, University of British Columbia, Vancouver, B.C., V6T 1Z4, Canada

[3] Crescenzi, P., Goldman, D., Papadimitriou, C., Piccolboni, A., Yanakakis, M. (1998). On the complexity of protein folding. *In*: Proceedings of the 13*th annual ACM symposium of theory of computing* (STOC 98) (p. 597–603)

[4] Lengauer, T. (1993). Algorithmic research problems in molecular bioinformatics. *In*: Proceedings of the second *Israel symposium on theory of computing systems (ISTCS)*, Natanya, Israel (p. 177–192).

[5] Richards, F. M. (1991). The protein folding problem. *Scientific American*, 264 (1) 54-7, 60-3.

[6] Chan, H. S., Dill, K. A. (1993). The protein folding problem. *Physics Today*, 46 (2), 24–32.

[7] Anfinsen, C. B., Haber, E., Sela, M., White, F. H. (1961). The kinetics of formation of native ribonuclease during oxidation of the reduced polypeptide chain. *In:* Proceedings of the *National Academy of Sciences*, 47 (9), 1309–1314.

[8] Unger, R., Moult, J. (1993). Genetic algorithms for protein folding simulations. *Journal of Molecular Biology*, 231, 75–81

[9] Dandekar, T., Argos, P. (1994). Folding the main chain of small proteins with genetic algorithm. *Journal of Molecular Biology*, 236, 844–861.

[10] Krasnogor, N., Hart,W. E., Smith, J. E., Pelta, D. A. (1999). Protein structure prediction with evolutionary algorithms. *In*: Proceedings of the 1999 *international genetic and evolutionary computation conference* (*GECCO*99), San Mateo CA (p. 1596–1601).

[11] Konig, R., Dandekar, T. (1999). Improving genetic algorithms for protein folding simulation by systematic crossover. *BioSystems*, 50, 17–25.

[12] Krasnogor, N., Blackburnem, B., Pelta, D. A., Burk, E. K. (2002). Multimeme algorithms for protein structure prediction. In *Lecture notes in computer science*. *In:* Proceedings of *parallel problem solving from nature* (2439, 769–778). Berlin: Springer.

[13] Pelta, D. A., Krasnogor, N. (2004). Multimeme algorithms using fuzzy logic based memes for protein structure prediction. *In*: Recent advances in memetic algorithms. Berlin: Springer.

[14] Cutello, V., Morelli, G., Nicosia, G., Pavone, M. (2005). Immune algorithms with aging operators for the string folding problem and the protein folding problem. In *Lecture notes in computer sciences* (3348, 80–90). Berlin: Springer.

[15] Cutello, V., Nicosia, G., Pavone, M., Timmis, J. (2006). An immune algorithm for protein structure prediction on lattice models. *IEEE Transaction on Evolutionary Computation.*

[16] Lesh, N., Mitzenmacher, M., Whitesides, S. (2003). A complete and effective move set for simple protein folding. *In*: Proceedings of the 7th *annual international conference on research in computational molecular biology* (*RECOMB*) (p. 188–195). New York: ACM Press.

[17] Jiang, T., Cui, Q., Shi, G., Ma, S. (2003). Protein folding simulations of the hydrophobic-hydrophilic model by combining tabu search with genetic algorithms. *Journal of Chemical Physics*, 119 (8) 4592–4596.

[18] Shmygelska, A., Hoos, H. H. (2003). An improved ant colony optimization algorithm for the 2D HP protein folding problem. In *Lecture notes in computer science*. *In:* Proceedings of *advances in artificial intelligence, AI* 2003 (p. 400–417). Berlin: Springer.

[19] Shmygelska, A., Hoos, H. H. (2005). An ant colony optimization algorithm for the 2D and 3D hydrophobic polar protein folding problem. *BMC Bioinformatics*, 6 (1) 30.

[20] Shmygelska, A., Hernandez, R., Hoos, H. H. (2002). An ant colony algorithm for the 2D HP protein folding problem. *In*: *Lecture notes in computer science. In:* Proceedings of the 3rd *workshop on ant algorithms* (2463, 40–52). Berlin: Springer.

[21] Dill, K. A., Bornberg, S., Yue, K., Fiebig, K., Yee, D., Thomas, P., Chan, H. (1995). Principales of protein folding-a perspective from simple exact models. Protein science 4, 561-602. Department of Pharmaceutical Chemistry, Box 1204, University of California, San Francisco, California 94143-1204

[22] Dill, K. A. (1985). Theory for the folding and stability of globular proteins. *Biochemistry*, 24 (6) 1501–1509.

[23] Dehouck Yves, (2005). Thèse de doctorat: Développement de potentiels statistiques pour l'étude in silico de protéines et analyse de structurations alternatives. Université libre de Bruxelles. Mai.

[24] Hussein A. Abbass. (2001). MBO: Marriage in honey Bees Optimization a Haplometrosi Polygynous Swarming Approach. *In*: Proceeding congress on evolutionary computation 2001 Seoul South Korea, 1, 207-214.

[25] Malek Rahoual. (2007). contribution a l'optimisation combinatoire mono et multi objectifs par des méthodes coopératives. Thèse de doctorat d'Etat en informatique. Université des sciences et de la technologie Houari Boumediene, département d'Informatique.

[26] Andrei B¢autu, Henri Luchian. (2010). Protein Structure Prediction in Lattice Models with Particle Swarm Optimization in ANTS, LNCS 6234, p. 512–519, Springer-Verlag Berlin Heidelberg.