2014 IEEE Congress on Evolutionary Computation (CEC)
July 6-11, 2014, Beijing, China

Protein Folding Estimation Using Paired-Bacteria Optimizer

Mengshi Li*, Tianyao Ji*, Peter Wu†, Shan He‡ and Qinghua Wu*
*School of Electric Power Engineering, South China University of Technology, Guangzhou, 510641, China
Email: wuqh@scut.edu.cn
†Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences
Institute of Biomedical and Health Engineering, Paul C. Lauterbur Research Center for Biomedical Imaging
Shenzhen, 518055, P. R. China.
‡School of Computer Science, University of Birmingham
Birmingham, B15 2TT, UK.

***Abstract*—Protein folding estimation attracts a large attention in the area of computational biology, due to its benefits on medical research and the challenge of NP-hard objective functions. In order to simulate the protein folding procedure and estimate the structure of the protein after folding, this paper adopts a Paired-Bacteria Optimizer (PBO), which is a biologically-inspired optimization algorithm. Compared with most Evolutionary Algorithms (EAs), the computational complexity of PBO is much less. Therefore, it is suitable to be applied to solve NP-hard problem. The experimental studies is performed on several benchmark lattice protein combination. The experimental results demonstrated that PBO is able to estimate the folded protein structure with a superior convergence.**

***Index Terms*—Protein folding, optimization, paired-bacteria optimizer.**

I. Introduction

Protein is a large biological molecule, which plays the fundamental role in life progress. The structure of protein consists of one or more chains of amino acid residues. With different combinations of the amino acid residues, proteins perform varieties of functions in living organisms [1]. The structures of protein are determined by the nucleotide sequence of their genes, and which usually results in folding of the protein into a specific three-dimensional structure that determines its behavior. The folding process reforms the protein from an unfolded polypeptide, when translated from a sequence of mRNA to a linear chain of amino acids, to a unique structure. This progress not only enhances the stability of the protein, but also enables its function [2]. Misfolded proteins cause variants of diseases, such as allergies and mad cow disease. Therefore, simulation of protein progress attracts great attention recently in the area of computational biology, which in turn further benefits the medical research in relevant areas.

In order to simulate protein folding, a hydrophobic-polar (H-P) model is introduced in this research, which simplifies the protein structure into two types of residues [3]. This model has been widely used in the study of computational biology, which aims to analyze the role of sub structures in protein. Several researches define the protein folding as an optimization problem, which aims to estimate the optimal structure that has the best stability in physics, *i.e.*, the H-P interactions have a minimum energy configuration. The problem of searching such an optimal structure is an NP-hard problem, which requires a great amount of evaluation with different trail structures. As the algorithm does not need to make any assumption on the objective function, heuristics algorithms show a great potential on solving NP-hard problem. Research [4] implements Genetic Algorithm (GA) with improved crossover and mutation operators. The experimental results have indicate that the heuristics algorithm outperforms conventional methods, such as Monte Carlo simulation. Research [5] indicates that the crossover operator is the major factor which limits the better performance of GA on the searching of optimal protein structure. Therefore, an Estimation of Distribution Algorithm (EDA) is improved from GA, which extends the selection procedure with an explicit probability [6]. Although the Markov probabilistic model in EDA enhances the algorithms searching ability, its computational complexity is increased as well.

Compared with EDA, Swarm-Intelligences (SIs), such as Particle Swarm Optimiser (PSO), not only has a small computational complexity, but also achieves superior performance on real number optimization problems. Therefore, this paper adopts an SI algorithm – PBO, which is inspired from the bacterial behaviors proposed in the Bacterial Foraging Algorithm with Varying Population (BFAVP) [7]. The previous research work has demonstrated that PBO has a superior performance than most EAs developed recently [7][8]. Followed by this study, PBO has also been applied to solve the optimal power flow problem with stochastic loads, and obtains a prospective result in our previous experiment [9]. Different from most EAs, PBO has only a pair of individuals in a population. The two individuals perform different tasks. The primary individual plays a role in data mark, which provides a reference for the next movement of the individuals. The associated individual performs random walk around the primary individual, which provides the gradient information for the primary individual. Occasionally, the associated individual is randomly placed at a location far away from the primary individual so as to prevent the premature results. In order to accelerate the convergence speed of the PBO, a simplified quorum sensing is introduced either attracting the primary individual to the

978-1-4799-1488-3/14/$31.00 ©2014 IEEE

best fitness location or exchanging the the searching history the two individuals have experienced.

The rest of the paper is organized as follows. Section II introduces the details of the mathematical models of PBO, and the implementation of the algorithm. Section III shows an H-P model to simulates the protein folding progress. This section also explains the procedure of the evaluation on the energy of a specified protein configuration. Section IV presents the simulation studies which are undertaken on several benchmark H-P models.

II. Paired-Bacteria Optimizer

PBO is a cutting-edge EA with a superior performance on expensive computational function. It is inspired from bacterial behaviours which consists of the pseudo gradient based searching and the quorum sensing scheme. Two mathematical models are presented in detail in this section respectively.

A. Pseudo Gradient Searching

The position of a bacterium at the k^{th} iteration is defined as $X^k = (x_1^k, x_2^k, ..., x_n^k) \in \mathbb{R}^n$, where x_i^k is the component of the X^k on the i^{th} dimension. Meanwhile, an associated bacterium $\widetilde{X}$ is associated to X at each iteration as a mutation of bacterium X. A dimension, $l,\ l \in \{1, 2, ..., n\}$, is randomly chosen for the mutation. The position of the associated bacterium $\widetilde{X}$ at the k^{th} iteration is given as:

$$\widetilde{X}^k = X^k + \delta D^k, \tag{1}$$

where $\delta D^k = (0, ..., 0, d_l^k, 0, ..., 0)$, and d_l^k is randomly chosen in the feasible space on the l^{th} dimension, which is denoted as:

$$d_l^k = c_1 r_1 (B_{\text{up}_l} - B_{\text{lo}_l}), \tag{2}$$

where $r_1 \in [-1, 1]$ is a random number, and B_{lo_l} and B_{up_l} denote the lower and upper boundary of the l^{th} dimension, respectively. A coefficient c_1, which has two values, is set to be 0.05 and 1. Equation (2) generates a random location around primary bacterium on the l^{th} dimension, and equation (1) places the associated bacterium on that location. By calculating a direction with better pseudo gradient using the fitness value and location of two individuals by equation (1), the algorithm will decide whether it will move to the primary individual to that direction.

Set $g_l^k(\widetilde{X}^k, X^k)$ in an alternative format of pseudo gradient along the l^{th} dimension at the k^{th} iteration, as follows.

$$g_l^k(\widetilde{X}^k, X^k) = \frac{F(\widetilde{X}^k) - F(X^k)}{\widetilde{x}_l^k - x_l^k}, \tag{3}$$

where $F(X^k)$ and $F(\widetilde{X}^k)$ are the evaluation values of X^k and $\widetilde{X}^k$, respectively. At each iteration, the velocity of the pseudo gradient searching is given as:

$$V_{\text{c}}^k = (0, ..., -g_l^k(\widetilde{X}^k, X^k), ..., 0). \tag{4}$$

B. Simplified Quorum Sensing

The velocity of the simplified quorum sensing is updated at the k^{th} iteration as follows,

$$V_{\text{q}}^k = r_2(P_{\text{g}}^k - X^k)), \tag{5}$$

where P_{g}^k indicates the position of the best bacterium from the past k iterations, and r_2 is a random number, $r_2 \in [0, 1]$. The l^{th} elements v_l^k in V^k is in the range of $[-v_l^{\max}, v_l^{\max}]$, where $v_l^{\max}$ is the maximal velocity of bacterium X along the l^{th} dimension, which is scaled proportionally to the range of the search boundary of that dimension. This equation is similar to social attraction in Particle Swarm Optimization (PSO) [10]. PBO employs this equation to accelerate the convergence in the early stage of the optimization process.

A dimensional velocity is inspired from the co-evolutionary algorithm, which assumes that the gradient along each dimension has similar tendency [11]. As a result, if the algorithm detects a better convergence along the l^{th} dimension, PBO will converge towards to that direction with a small step length on all dimensions. According to out previous research, dimensional velocity increases the convergence speed in the early stage [12]. The dimensional velocity of primary individual movement is updated as:

$$V_{\text{s}}^k = \begin{cases} r_3(X^k \ominus \widetilde{x}_l^k), & F(\widetilde{X}^k) < F(X^k), \\ 0, & \text{otherwise}, \end{cases} \tag{6}$$

where $r_3 \in [0, 1]$ is a uniformly distributed random number, and $\mathbf{a} \ominus b$ means a constant b ($b \in \mathbb{R}$) is subtracted from each element $\mathbf{a}$.

The velocity of the primary bacterium at the k^{th} iteration is donated by V^k, $V^k \in \mathbb{R}^n$, $V^k = (v_1^k, v_2^k, ..., v_n^k)$, where v_i^k is a component of the V^k on the i^{th} dimension. The velocity of the primary bacterium combines the the speed in pseudo gradient searching and simplified quorum sensing, which is expressed as:

$$V^k = V_{\text{c}}^k + V_{\text{q}}^k + V_{\text{s}}^k \tag{7}$$

Hence, the position of the primary bacterium is updated as:

$$X^{k+1} = X^k + \gamma V^k. \tag{8}$$

where γ indicates the inertia weight, which is fixed in each iteration. In an optimization process, pseudo gradient searching and simplified quorum sensing are performed iteratively. The pseudo code of PBO is listed in Table I.

III. Evaluation of Protein Folding

A. H-P Models

Currently, there is not any correct model to accurately describe how the folding is performed in sequence. In classical view, the folding progress has a specified pathway, which consists of several steps. The energy is minimized through the pathway step by step. However, research [6] proposes a new folding model, which assumes the folding is a random progress without a fixed pathway. The energy can be both increased or decreased during the folding progress. When the protein has a stable structure with minimized energy, the folding progress is

TABLE I
PSEUDO CODE OF PBO

Initialize the position of the primary individual and pseudo individual;
Initialize the velocity of the primary individual;
Evaluate the fitness of the primary individual;
Set g_p^0 as the guessed value of global optimum;
Set $k := 1$;
WHILE(the termination conditions are not met)
Select a dimension l randomly;

Pseudo individual:	Place the pseudo individual to position $\widetilde{X}^k$ by equation (1); Evaluate the fitness value of $\widetilde{X}^k$;
Gradient updating:	Exchange the gradient information on each dimension by equation (6);
Velocity updating:	Calculate the velocity of primary individual by equation (7);
Generation:	Update the position of the primary individual by equation (8); Update g_p^k if the fitness of current individual is better;

$k := k + 1$
END WHILE

complete. In the new view, the searching of the folded structure is a high dimensional multimodal optimization problem, which has a large number of local optima.

Before the folding, polypeptide is a single linear polymer chain derived from the condensation of amino acids. Over twenty types of amino acids are found inside the cell, which are the fundamental materiel to compose the protein. However, simulating the folding progress of amino acids in 3 dimensional space is too complex to evaluate the algorithm, as the implementation of the model for amino acid chain requires a large number of predefined chemical settings. Therefore, this research adopts a simplified H-P model to describe the protein structure. In this model, H indicates a hydrophobic residue, which is the dominant force in the protein folding. On the other hand, P indicates a polar residue, which connects hydrophobic residue and maintains the protein structure. The H-P model is considered as a suitable benchmark for the studies of computational biology and optimization.

The energy function, which is defined as the objective function in the optimization progress, is calculated from the topological structure of the H-P model. The optimization aims to minimize the total energy of the H-P model by evaluating the energy of topological neighbours in the space. Different topological neighbours of the residues in the space have different energy levels, which are expressed as:

$$\begin{aligned} E_{\mathrm{HH}} &= -1, \\ E_{\mathrm{HP}} &= 0, \\ E_{\mathrm{PP}} &= 0, \end{aligned} \tag{9}$$

where E_{HH}, E_{HP} and E_{PP} are the energy measured of H-H, H-P and P- P topological neighbours. Figure 1 shows the configuration of a protein with the sequence of HHHPHH-HHPHP. In this figure, the dot indicates an H residue and the circle indicates a P residue. It can be found that there are two H-H topological neighbours (marked as dashed line), one P-P topological neighbour (marked also in dotted line), and one H-P topological neighbour (marked as dotted line). Therefore, the total energy of this configuration is calculated as:

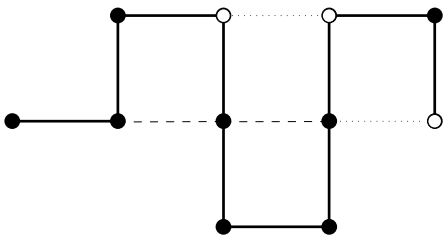

Fig. 1. A configuration of the H-P model folded structure.

$$E = 2 \cdot E_{\mathrm{HH}} = -2. \tag{10}$$

B. Trial Protein Sequence Evaluation

Before the evaluation of the trial protein sequence, two two-dimensional matrices, Y and Z, are initialized to record the folded protein structure and protein placement sequence. Assuming the objective protein has n residues, the size of the matrices Y and Z will be set to $(2n-1) \times (2n-1)$. Element $Y(j,k)$ indicates the type of the residue, which is defined as:

$$Y(j,k) = \begin{cases} 0 & \text{space} \\ 1 & \mathrm{H-residue} \\ 2 & \mathrm{P-residue.} \end{cases} \tag{11}$$

Element $Z(j,k)$ indicates the sequence number of the residue placed at the coordinate (j,k). If the i^{th} residue of the protein is placed at coordinate (j,k), $Z(j,k)$ will be set to i. A zero value in Z indicates an empty space, meaning no residue is placed in that location.

The first residue is placed at the coordinate of (n,n). As a result, $Z(n,n)$ is set to 1. In this experiment, the trial solution X, discussed in Section II, indicates a folding sequence of the protein combination. The i^{th} element of X, x_i, determines the folding direction of the $(i+1)^{\mathrm{th}}$ residue, which is expressed as:

$$x_i \in \begin{cases} [0,1] & \text{rotate } 90^\circ \text{ left;} \\ (1,2] & \text{rotate } 90^\circ \text{ right;} \\ (2,3] & \text{rotate } 0^\circ. \end{cases} \tag{12}$$

Figure 2 shows an example of the residue placement operation. The sub-figure *a* indicates a 90° left rotation placement, where $x_i \in [0,1]$. The solid line shows the original residue placement direction. Meanwhile, sub-figure *b* indicates a 90° right rotation placement, where $x_i \in (1,2]$. When $x_i \in (2,3]$, the $(i+1)^{\mathrm{th}}$ residue will be placed along the original direction as demonstrated in sub figure *c*.

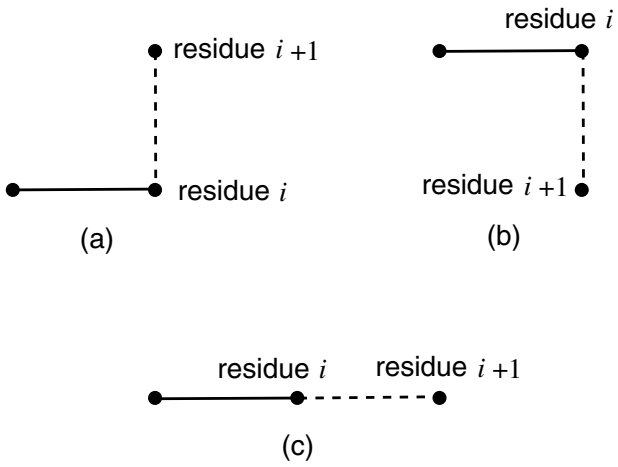

Fig. 2. An example of the residue placement operation.

After all residues are placed, the energy configuration is calculated using an energy matrix G:

$$G_{jk} = \begin{cases} -2 & Y_{jk} = 1,\ Y_{(j+1)k} = 1 \text{ and } Y_{j(k+1)} = 1 \\ -1 & Y_{jk} = 1 \text{ and } Y_{(j+1)k} = 1 \\ -1 & Y_{jk} = 1 \text{ and } Y_{j(k+1)} = 1 \\ 0 & \text{otherwise} \end{cases}, \quad (13)$$

Thus, the total energy configuration is calculated as:

$$E = \sum_{i=1}^{2n} \sum_{j=1}^{2n} G_{ij}, \quad (14)$$

The placement sequence is randomly generated by PBO, and overlap of the residue may happen during the placement. Therefore, a penalty function is introduced to calculate the missing residues after the placement. Assume there are m elements in matrix Y, which have non-zero values. The number of the missing residue is $n - m$. Thus, the objective function of the protein folding is rewritten as:

$$E = \sum_{i=1}^{2n} \sum_{j=1}^{2n} G_{ij} + \lambda_{\mathrm{p}}(n - m), \quad (15)$$

where λ_{p} represents the coefficient for the penalty function. λ_{p} is set to 100 in this study.

IV. Experimental Studies

A. Experimental Setting

PBO was evaluated in comparison with PSO [10]. PSO is a Swarm Intelligence Algorithm (SIA), which has been widely studied and compared in the past few years. The implementation of PSO was described in detail in [10]. For the parameters of PSO, the inertia weight ω is set to 0.73, and the acceleration factors c_1 and c_2 are both set to 2.05. The number of iterations for PSO to simulate protein folding is set to 1.0×10^4, and the population size is set to 100. Therefore, the total number of function evaluations for PSO is 1.0×10^6, which has the same value as the setting of PBO. As there are only 2 individuals, the iterations of PBO is set to be 5.0×10^5.

TABLE III
The number of residues, minimal folded energy and the sequence of the H-P instances

Instant	Optimum	Algorithm	Minimum	Mean
f_1	-9	PBO	-9	-8.78
		PSO	-9	-8.68
f_2	-14	PBO	-14	-13.42
		PSO	-14	-13.16
f_3	-21	PBO	-21	-19.38
		PSO	-20	-18.58
f_4	-42	PBO	-38	-36.5
		PSO	-35	-33.36
f_5	-48	PBO	-42	-38.84
		PSO	-38	-32.64

In the experimental study, a set of H-P instances has been adopted to evaluate the folding performance of the proposed algorithm. These H-P instances are adopted in [6] and [13]. Table II lists the number of residues, minimal folded energy and the sequence of these instances. The sequences of H-P instances are specially designed, which follows the natural protein behaviors. There is only one explicit structure, which has the minimal folding energy, for each H-P instance. The numbers of residues of these proteins range from 20 to 100. Therefore, the folding simulations are high-dimensional multi-modal optimization problem, which is suitable to be solved using PBO.

B. Experimental Results

During the experiment, PBO and PSO are applied to minimize the folded energy for the benchmark H-P instances. Each algorithm is applied to these benchmark functions with 50 individual runs. Table III compares the minimal and mean energy of the configuration obtained by PBO and PSO. The overall values of the average energy in the table demonstrates that PBO outperforms PSO on all these 5 benchmark functions. For the instances with simple structures, such as f_1 and f_2, both the two algorithms successfully estimate the optimal folded structure, which has the minimal energy as known. PSO fails to find the optimal configuration when the number of residues is increased to 50, whereas PBO still shows a great searching potential. However, the results on the complex cases, such as f_4 and f_5, show that PSO and PBO are not able to locate the optimal structure when the number of residues is increased over 64. Figure 3 shows an optimal configuration obtained by PBO on instance f_1.

Figure 4 illustrates the convergence curves of PSO and PBO on f_1. It can be found that PSO converges rapidly in the early period of searching, which is caused by the large population size. As there are only two individuals, the convergence speed of PBO is slightly lower. However, PBO outperforms PSO on the overall period of the searching. The objective function of the protein folding is an NP-hard problem with a large number of local optima. Therefore, the algorithms that have limited mutation operators, such as PSO, are more likely to be trapped in local optima. The convergence curve of PBO indicates that the gradient searching in the algorithm overcomes this

TABLE II
THE NUMBER OF RESIDUES, MINIMAL FOLDED ENERGY AND THE SEQUENCE OF THE H-P INSTANCES

Instant	umber of residues	Optimal folded energy	Sequence
f_1	20	-9	$\{HP\}^2P\{H^2P\}^2HPH^2P^2HPH$
f_2	36	-14	$P^3\{H^2P^2\}^2P^3H^7P^2H^2P^4H\{HPP\}^2$
f_3	50	-21	$H\{HP\}^4H^4PH\{P^3H\}^2P^4H\{P^3H\}^2PH^4\{PH\}^4H$
f_4	64	-42	$H^{12}\{PH\}^2\{P^2H^2\}^2P^2H\{P^2H^2\}^2P^2H\{P^2H^2\}^2P^2\{PH\}^3H^{11}$
f_5	99	-48	$P^6HPH^2P^5H^3PH^5PH^2P^4H^2P^2H^2\{PH^5\}^2H^5\{PH^2\}^2H^5P^11H^7P\{PH\}^2H^2P^6HPH$

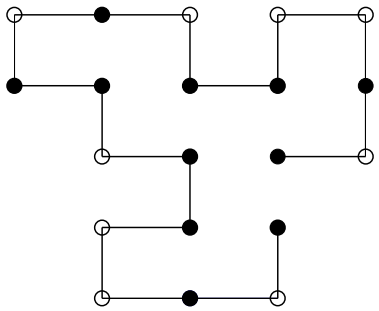

Fig. 3. An example of the optimial configuration obtained by PBO on instance f_1.

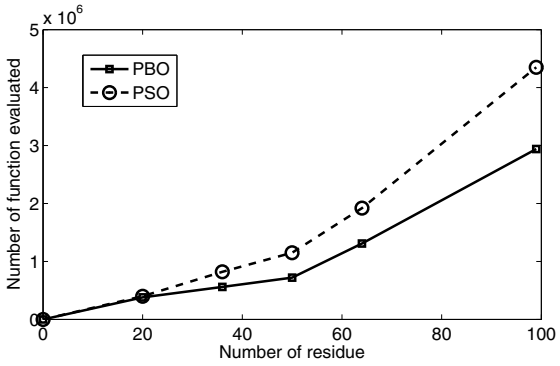

Fig. 5. Computational complexity comparison between PBO and PSO.

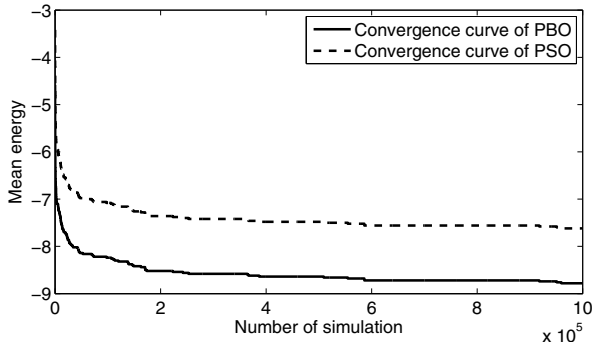

Fig. 4. Convergence curve of PBO and PSO on instance f_1.

drawback, which allows the individuals to escape from the local optima.

Folding simulations on large-scale protein structure are necessary due to its applications in the area of medicine research. Therefore, another critical issue of this study is the comparison on computation complexities of the adopt algorithms. In this experiment, a threshold, which has the value of 80% of the optimal energy, is set for each benchmark instance. Once the algorithms obtain a optimised folded structure, which has a energy less than the threshold, the number of the function evaluated will be recorded. Figure 5 illustrates the computational complexity comparison between PBO and PSO. The horizontal axis indicates the number of the residues in the protein, and the vertical axis indicates the function evaluated to reach the threshold. These two curves demonstrated that PBO is more suitable to be applied to the folding simulation with complex structure. With the increasing of the number of residues in the protein, PBO consumes lesser and lesser computational time during the optimization.

V. CONCLUSION

This paper presents a method to evaluate the energy value of a configuration on a folded protein. Based on this method, optimization algorithms are able to be applied to optimise the protein folded structure when minimizing the energy value. Meanwhile, a novel bacteria-inspired optimization algorithm, PBO, is adopted to solve this optimization, which has merits of fast convergence and easy implementation. The performance of PBO on protein folding is compared with PSO. The experimental results have demonstrated that with the same energy configuration of the folded protein structure, the time consumption of the searching procedure of PBO is much less than that of PSO.

VI. ACKNOWLEDGEMENTS

This research is jointly supported by Guangdong Natural Science Foundation (No. S2013040016964) and Guangdong Innovative Research Team Program(No. 201001N0104744201).

REFERENCES

[1] Christopher M Dobson. Principles of protein folding, misfolding and aggregation. *Seminars in Cell & Developmental Biology*, 15(1):3 – 16, 2004. ¡ce:title¿Protein Misfolding and Human Disease and Developmental Biology of the Retina¡/ce:title¿.

[2] Robert B. Freedman, Tim R. Hirst, and Mick F. Tuite. Protein disulphide isomerase: building bridges in protein folding. *Trends in Biochemical Sciences*, 19(8):331 – 336, 1994.

[3] Bonnie Berger and et al. Protein folding in the hydrophobic-hydrophilic (hp) model is np-complete. 1998.

[4] Ron Unger and John Moult. Genetic algorithm for 3d protein folding simulations. In *Proceedings of the 5th International Conference on Genetic Algorithms*, pages 581–588, San Francisco, CA, USA, 1993. Morgan Kaufmann Publishers Inc.

[5] N. Mansour, F. Kanj, and H. Khachfe. Evolutionary algorithm for protein structure prediction. In *Natural Computation (ICNC), 2010 Sixth International Conference on*, volume 8, pages 3974–3977, 2010.

[6] R. Santana, P. Larranaga, and J.A. Lozano. Protein folding in simplified models with estimation of distribution algorithms. *Evolutionary Computation, IEEE Transactions on*, 12(4):418–438, 2008.

[7] M. S. Li, T. Y. Ji, W. J. Tang, and Q. H. Wu. Bacterial foraging algorithms with varying population. *BioSystems*, 100:185–197, June 2010.

[8] M. S. Li, W. J. Tang, Q. H. Wu, and J. R. Saunders. Paired-bacteria optimiser - a simple and fast algorithm. *Inf. Process. Lett.*, 111:809–813, August 2011.

[9] M. S. Li, T. Y. Ji, Q. H. Wu, and Y. S. Xue. Stochastic optimal power flow using a paired-bacteria optimizer. In *Power and Energy Society General Meeting, 2010 IEEE*, pages 1 –7, july 2010.

[10] J. Kennedy and R.C. Eberhart. Particle swarm optimization. volume 4, pages 1942–1948, Perth, Australia, 1995.

[11] Z. Yang, K. Tang, and X. Yao. Large scale evolutionary optimization using cooperative coevolution. *Information Sciences*, 178(15):2985–2999, 2008.

[12] H. L. Liao Q. H. Wu M. S. Li, T. Y. Ji. Optimal power flow with environmental constraint using a paired bacterial optimiser. In *Accepted by Power Engineering Society General Meeting, 2004. IEEE*, 2011.

[13] Alena Shmygelska, Rosala Aguirre-Hernndez, and HolgerH. Hoos. An ant colony optimization algorithm for the 2d hp protein folding problem. In Marco Dorigo, Gianni Caro, and Michael Sampels, editors, *Ant Algorithms*, volume 2463 of *Lecture Notes in Computer Science*, pages 40–52. Springer Berlin Heidelberg, 2002.