Expert Systems with Applications 36 (2009) 12446–12453

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

An efficient hybrid Taguchi-genetic algorithm for protein folding simulation

Cheng-Jian Lin *, Ming-Hua Hsieh

Department of Computer Science and Information Engineering, National Chin-Yi University of Technology, Taichung County 411, Taiwan, ROC

ARTICLE INFO

Keywords:
Protein structure prediction
HP lattice model
Genetic algorithm
Taguchi method
Particle swarm optimization

ABSTRACT

Given the amino-acid sequence of a protein, the prediction of a protein's tertiary structure is known as the protein folding problem. The protein folding problem in the hydrophobic–hydrophilic lattice model is to find the lowest energy conformation. In order to enhance the performance of predicting protein structure, in this paper we propose an efficient hybrid Taguchi-genetic algorithm that combines genetic algorithm, Taguchi method, and particle swarm optimization (PSO). The GA has the capability of powerful global exploration, while the Taguchi method can exploit the optimum offspring. In addition, we present the PSO inspired by a mutation mechanism in a genetic algorithm. We demonstrate that our algorithm can be applied successfully to the protein folding problem based on the hydrophobic-hydrophilic lattice model. Simulation results indicate that our approach performs very well against existing evolutionary algorithm.

1. Introduction

The prediction of protein structure from its amino-acid sequence is one of the most prominent problems in computational biology. A protein's function depends mainly on its tertiary structure, which in turn depends on its primary structure. Mistakes in the folding process create proteins with abnormal shapes, which are the causes of diseases such as cystic fibrosis, Alzheimer's, and mad cow. If we could predict the tertiary structures of proteins from their sequences, we would be able to treat these diseases better. The knowledge of protein tertiary structures also has other applications, such as in the structure-based drug design field (Bui & Sundarraj, 2005).

Currently, protein structures are primarily determined by techniques such as MRI (magnetic resonance imaging) and X-ray crystallography, which are expensive in terms of equipment, computation, and time. Additionally, these techniques require isolation, purification, and crystallization of the target protein. Computational approaches to protein structure prediction are therefore very attractive. The difficulty in solving protein structure prediction problems stems from two major sources: (1) finding good measures for the quality of candidate structures, and (2) given such measures, determining optimal or close-to-optimal structures for a given amino-acid sequence (Krasnogor, Hart, Smith, & Pelta, 1999).

Recently, many researchers (Krasnogor, Pelta, Lopez, Mocciola, & de la Canal, 1998; Krasnogor et al., 1999; Patton, Punch, III, & Goodman, 1995; Pedersen & Moukt, 1997) have used evolutionary algorithms, such as the genetic algorithms (GA), for solving the protein folding problem. Genetic algorithms are stochastic search techniques based on the mechanism of natural selection, which requires information to search effectively in a large or poorly understood search space. The effectiveness of crossover and mutation is weakened in the protein folding problem (Krasnogor et al., 1998, 1999), since by increasing the compact folded structure, the failure of the crossover operation increases due to collisions. Further, in sequences of mutation, there will be often invalid conformations due to collisions within compact conformation. Therefore, some researchers (Bui & Sundarraj, 2005; Jiang, Cui, Shi, & Ma, 2003; König & Dandekar, 1999; Takahashi, Kita, & Kobayashi, 1999) have proposed various hybrid methods to improve GA. The above-mentioned improved GA methods were mainly aimed at the crossover and mutation operations.

Recently, the Taguchi method is a robust design approach. It uses many ideas from statistical experimental design for evaluating and implementing improvements in products, processes, and equipment. The fundamental principle is to improve the quality of a product by minimizing the effect of the causes of variation without eliminating the causes. The Taguchi method is suitable for a wide range of applications (Kaytakoğlu & Akyalçın, 2007; Liu, Fung, & Wang, 2007; Wang & Huang, 2008), including the following practices: quality engineering, experimental design, business data analysis, management by total results, pattern recognition, and so on. The Taguchi method is a series of approaches that predicts and prevents troubles or problems that might occur in the market after a product is sold and used by a

* Corresponding author.
E-mail address: cjlin@ncut.edu.tw (C.-J. Lin).

0957-4174/$ - see front matter Crown Copyright © 2009 Published by Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2009.04.074

customer under various environment and real-life conditions for the duration of the product life.

In this paper, we focus on the 2D hydrophobic-polar (HP) lattice model. An efficient hybrid Taguchi-genetic algorithm (HTGA) is proposed for solving the protein folding problem in the 2D HP model. The proposed HTGA is mainly aimed to improve the crossover and mutation operators and enhance exploitation capability. In order to improve the crossover operation, we use the Taguchi method to select the better genes. In the mutation operation, we employ the merits of PSO to improve the mutation mechanism. Lin, Liu, and Lee (2008) and Lin and Hong (2007) used the particle swarm optimization to improve the mutation mechanism. Simulation results show that our method has a better performance than those of existing methods in protein folding problem.

The remainder of this paper is structured as follows: Section 2 gives the preliminaries and the formal definition of the protein folding problem in the 2D HP lattice model. Section 3 describes our approach in detail. The proposed hybrid Taguchi-genetic algorithm combining the traditional genetic algorithm, the Taguchi method, and particle swarm optimization is presented. The experimental results obtained by our method and by other methods are compared in Section 4. Finally, the conclusion is given in the last section.

2. Preliminaries

In this section, we briefly present the 2D HP protein folding problem and its free energy calculation.

2.1. The 2D HP protein folding problem

Lattice proteins are highly simplified computer models of proteins which are used to investigate protein folding. Dill (1985) proposes the hydrophobic-polar model. Because proteins are such large molecules, containing hundreds or thousands of atoms, it is not possible with current technology to simulate more than a few microseconds of their behavior in complete atomic detail. Hence, real proteins cannot be folded in a computer simulation. Lattice proteins, however, are simplified into two ways: the amino acids are modeled as single "beads" rather than by every atom, and the beads are restricted to a rigid (usually cubic) lattice. This simplification means they can fold to their energy minima in a time quick enough to be simulated. Lattice proteins are made to resemble real proteins by introducing an energy function (Takahashi et al., 1999), a set of conditions which specify the energy of interaction between neighboring beads, usually taken to be those occupying adjacent lattice sites. The energy function mimics the interactions, which include hydrophobic and hydrogen bonding effects, between amino acids in real proteins. The beads are divided into types, and the energy function specifies the interactions, depending on the bead type, just as different types of amino acid interact differently.

One of the most popular lattice models, the HP model, feature just two bead types: H (hydrophobic or non-polar) and P (hydrophilic or polar). An instance is shown in Fig. 1 for the 2D HP lattice model (Guo, Feng, & Wang, 2007). The black squares denote the hydrophobic amino acid and the white squares denote the hydrophilic. The dotted line denotes the H–H contacts (free energy) in the conformation, which are assigned an energy value of −1. The free energy is minimum value; the number of H–H contact is the maximum. Fig. 1 shows a protein structure with 9 H–H contacts (energy = −9). Since the native state of a protein generally corresponds to the lowest free energy state for the protein, the optimal conformation in the HP model is the one that

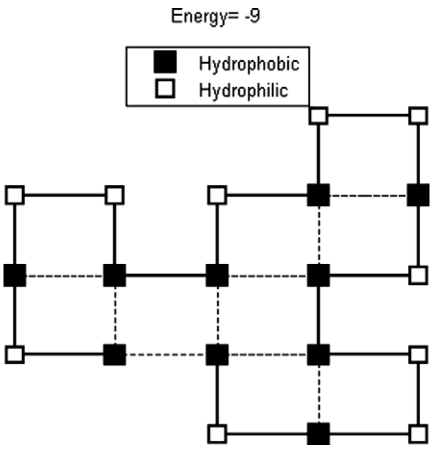

Fig. 1. An optimal conformation for the sequence "$(HP)_2PH(HP)_2(PH)_2HP(PH)_2$" in a 2D lattice model.

has the maximum number of H–H contacts which gives the lowest energy value.

2.2. Calculating the free energy

For any sequence in any particular structure, the free energy can be rapidly calculated from the free energy function. For the simple HP model, this is simply an enumeration of all the contacts between the H residues that are adjacent in the structure but not in the chain. Most researchers consider a lattice protein sequence protein-like only if it possesses a single structure with an energetic state lower than in any other structure. This is the energetic ground state, or the native state. The relative positions of the beads in the native state constitute the lattice protein's tertiary structure. Lattice proteins do not have genuine secondary structure, although some researchers have claimed that they can be extrapolated to real protein structures, which do include secondary structure, by appealing to the same law by which the phase diagrams of different substances can be scaled onto one another. By varying the free energy function and the bead sequence of the chain (the primary structure), effects on the native state structure and the kinetics (rate) of folding can be explored. This may provide insights into the folding of real proteins. In particular, lattice models have been used to investigate the free energy landscapes of proteins (Guo et al., 2007), i.e. the variation of their internal free energy as a function of conformation. We present the minimum free energy function of the 2D HP lattice model with calculation conditions as follows:

$$E^* = \left[\frac{\sum_{i=1}^{n} \left[\sum_{j=1}^{n} f(i) S(i,j) \right]}{2} \right] \quad n = \text{length of the protein sequence} \tag{1}$$

$$\text{s.t. } S(i,j) = f(j) * \|d_j - d_i\| = 1 \quad \text{and} \quad \|i-j\| > 2$$

where f is a mapping function: $f \to \{0,1\}$. That is, $f(i) = 0$ represents the hydrophilic residue and $f(i) = 1$ represents the hydrophobic residue. $\boldsymbol{d} = \{d_1, d_2, \ldots, d_j, \ldots, d_n\}$ is a vector set, where d_j denotes a projection onto the Cartesian coordinate. If each residue is connected to its sequence neighbor on an adjacent lattice site, then $S(i,j) = 1$. Otherwise $S(i,j) \neq 1$. Each lattice site is only occupied by one amino acid residue, which we call a conformation valid.

3. Methods

In this section, we review the Taguchi method and particle swarm optimization. An efficient hybrid Taguchi-genetic algorithm is also presented.

3.1. Review of the Taguchi method

The Taguchi method is a robust design approach developed by Taguchi, Chowdhury, and Taguchi (2000) that uses many ideas from statistical experimental design for evaluating and implementing improvements in products, processes, and equipment. The foundation of an experimental design is a method for efficiently designing experiments and analyzing the results. It searches for cause-and-effect relationships to improve quality by minimizing the effect of the causes of variation without eliminating the causes (Ross, 1989; Roy, 1990; Taguchi et al., 2000; Wu & Wu, 2000). Taguchi's parameter design uses two major tools, the orthogonal array (OA) and the signal-to-noise ratio (SNR). The SNR measures quality, and orthogonal arrays are used to study many design parameters simultaneously.

In order to explore the design space, factors are assigned an orthogonal array. The weakness of a typical shotgun approach or the one-factor-at-a-time method is that conclusions are drawn from a fixed design configuration. An orthogonal array provides a balanced set of experimentation runs such that the conclusions are drawn in a balanced fashion. The general symbol for two-level standard orthogonal arrays is $L_n(2^{n-1})$, where $n = 2^k$ is number of experiment runs; k denotes a positive integer which is greater than one; the symbol 2 is the number of levels for each factor; and $n-1$ denotes the number of columns in the orthogonal array.

Table 1 is an example of an orthogonal array, called the L_8 array. In this table, factors A, B,...,G are assigned to columns 1, 2,...,7, respectively. It is used to design experiments involving up to seven 2 level factors. Each column contains four level 1 and four level 2 conditions for the factor assigned to the column. Two 2 level factors combine in four possible ways, such as (1,1), (1,2), (2,1), and (2,2). Thus, all seven columns of an L (Latin) are orthogonal to each other. The array forces all experimenters to design almost identical experiments. Experimenters may select different designations for the columns but the eight trial runs will include all combinations independent of column definition. Thus, the OA assures consistency of design by different experimenters.

The signal-to-noise ratio is a quality index that has been historically used in the communications industry to evaluate communications systems. In quality engineering, the concept of SNR has been adapted by Genichi Taguchi to evaluate the quality of a product or a manufacturing process. The SNR is an index of robustness since it measures the quality of energy transformation that occurs within a design. The quality of energy transformation is expressed as the ratio of the level of performance of the desired function to the variability of the desired function. A high ratio represents high quality. There are several categories of SNR, depending on the type of characteristic: lower is better (LB), nominal is best (NB), and higher is better (HB) (Ross, 1989).

The SNR, which condenses multiple data points within a trial, depends on the type of characteristic being evaluated. The equations for calculating SNR (η) for LB and HB characteristics are:

Table 1
Orthogonal array L_8 (2^7).

Experiment number	Factors						
	A	B	C	D	E	F	G
1	1	1	1	1	1	1	1
2	1	1	1	2	2	2	2
3	1	2	2	1	1	2	2
4	1	2	2	2	2	1	1
5	2	1	2	1	2	1	2
6	2	1	2	2	1	2	1
7	2	2	1	1	2	2	1
8	2	2	1	2	1	1	2

(i) Lower is better (LB)

$$\eta_{\mathrm{LB}} = -10\log\left(\frac{1}{n}\sum_{i=1}^{n} y_i^2\right) \tag{2}$$

where n is the number of tests in an experiment; we have a set of characteristics y_i.

(ii) Higher is better (HB)

$$\eta_{\mathrm{HB}} = -10\log\left(\frac{1}{n}\sum_{i=1}^{n} \frac{1}{y_i^2}\right) \tag{3}$$

which is also measured in decibels.

An orthogonal array is used for optimization, i.e., to maximize the signal-to-noise ratio. It is necessary to use an orthogonal array and instead use the signal-to-noise ratio as the most important evaluation criteria.

3.2. Review of particle swarm optimization

Particle swarm optimization was originally introduced by Kennedy and Eberhart in 1995 to study social and cognitive behavior (Eberhart & Kennedy, 1995a, 1995b). The idea originated in studies on the synchronous flocking of birds and the schooling of fish. The PSO has come to be widely used as a problem solving method in engineering and computer science. This algorithm has several highly desirable attributes, including a basic algorithm that is very easy to understand and implement. It is similar in some ways to evolutionary algorithms, but requires less computational bookkeeping and generally fewer lines of code.

In the PSO, the trajectory of each individual in the search space is adjusted by dynamically altering the velocity of each particle, according to its own flight experience and the flight experience of the other particles in the search space. The position vector and the velocity vector of the ith particle in the D-dimensional search space can be represented by $X_i = (x_{i1}, x_{i2}, x_{i3}, \ldots, x_{id})$ and $V_i = (v_{i1}, v_{i2}, v_{i3}, \ldots, v_{id})$, respectively. According to a user-defined fitness function, suppose that the best position of each particle (which corresponds to the best fitness value obtained by that particle at time) is $P_i = (p_{i1}, p_{i2}, p_{i3}, \ldots, p_{id})$, and the fittest particle found so far is $P_g = (p_{g1}, p_{g2}, p_{g3}, \ldots, p_{gd})$. Then the new velocities and the positions of the particles for the next fitness evaluation are calculated using the following two equations:

$$v_{id}^{k+1} = v_{id}^{k} + c_1 \times rand(\cdot) \times (P_{id} - x_{id}^{k}) + c_2 \times rand(\cdot) \times (P_{gd} - x_{id}^{k}) \tag{4}$$

$$x_{id}^{k+1} = x_{id}^{k} + v_{id}^{k+1} \tag{5}$$

where c_1 and c_2 are constants known as acceleration coefficients. The $rand(\cdot)$ is uniformly distributed random numbers in the range [0,1].

3.3. The proposed hybrid Taguchi-genetic algorithm (HTGA)

Fig. 2 shows the flowchart of the proposed hybrid Taguchi-genetic algorithm (HTGA). The HTGA method works with a population of candidate solutions. At each generation, the n best individuals of the population are selected based on their fitness (the minimum free energy). The details are illustrated as follows:

3.3.1. Initialization step

If the input amino acid sequence is of length n, then each individual in the population is a string of length $n-1$ over the symbols = $\{U, L, R, D\}$, and that denotes a valid conformation in the 2D square lattice (Yap & Cosic, 1999). The symbols U, L, R and D are used to denote the fold directions *up*, *left*, *right* and *down* in

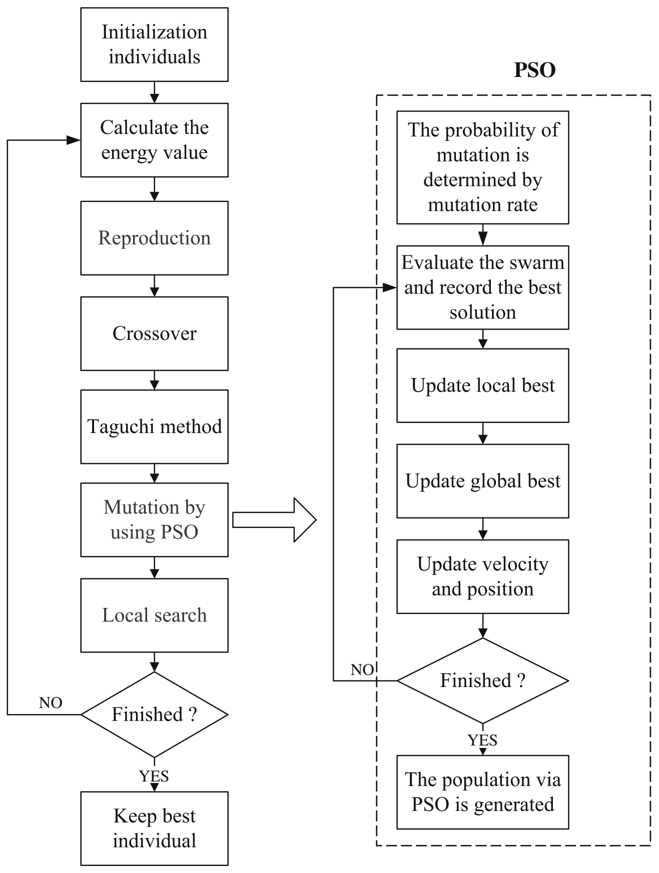
Initialization individuals
Calculate the energy value
Reproduction
Crossover
Taguchi method
Mutation by using PSO
Local search
Finished ?
NO
YES
Keep best individual
PSO
The probability of mutation is determined by mutation rate
Evaluate the swarm and record the best solution
Update local best
Update global best
Update velocity and position
Finished ?
NO
YES
The population via PSO is generated

Fig. 2. The proposed hybrid Taguchi-genetic algorithm for protein folding problem.

the encoding scheme, respectively. An initial population is generated randomly and initializes an $n-1$ dimensional space within a fixed range.

3.3.2. Reproduction step

Reproduction is a process in which individual strings are copied according to their fitness value. In this study, we use the roulette-wheel selection method (Cordon, Herrera, Hoffmann, & Magdalena, 2001)—a simulated roulette is spun—for this reproduction process. The best performing individuals in the top half of the population (Lin & Hsu, 2007) advances to the next generation. The other half is generated to perform crossover and mutation operations on individuals in the top half of the parent generation.

3.3.3. Crossover step

Reproduction directs the search toward the best existing individuals but does not create any new individuals. In nature, an offspring has two parents and inherits genes from both. The main operation working on the parents is the crossover operation, the operation of which occurred for a selected pair with a crossover rate that was set to 0.8 in this study. The first step is to select the individuals from the population for the crossover. Tournament selection (Cordon et al., 2001) is used to select the top-half of the best performing individuals (Lin & Hsu, 2007). The individuals are crossed and separated using a two-point crossover operation.

In Fig. 3, new individuals are created by exchanging the site's direction between i and j point (the selected sites of individual's parents). After this operation, the individuals with poor performances are replaced by the newly produced offspring.

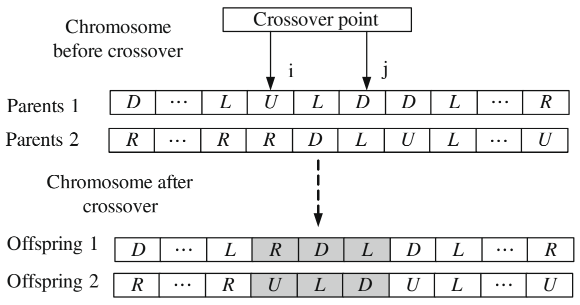

Fig. 3. Two-point crossover operation between *i* and *j* chromosomes.

3.3.4. Taguchi step

After the crossover operation, we randomly choose two chromosomes from each run to execute the orthogonal array experiment. The details are illustrated as follows (see Fig. 4):

In this study, we use a two-level orthogonal array. There are Q factors, where Q is equal to the length of the input amino acid sequence minus one. We select the appropriate orthogonal array of Q factors with two levels. Let $L_n(2^{n-1})$ represent $n-1$ columns and n individual experiments correspond to the n rows, where $n = 2^k$, k = a positive integer, and $Q \leqslant n-1$. If $Q < n-1$, only Q columns are used. We delete the last $(n-1)-Q$ columns to get an orthogonal array with Q factors.

We modified the SNR (η) of Eqs. (2) and (3) for the protein folding problem. Let $\eta_i = (y_i)^2$, where y_i is the energy function evaluation value of experiment i, where $i = 1, 2, \ldots, n$ experiments. The effects of the various factors can be defined as follows:

$$f_l = \text{total sum of } \eta_i \text{ for factor } f \text{ at level } l \qquad (6)$$

where i is the experiment number, f is the folding direction, and l is the level number.

The orthogonal arrays of the Taguchi method are used to study a large number of decision variables with a small number of experiments. An orthogonal array is used for optimization, i.e., to maximize the signal-to-noise ratio. However, it is necessary to use an orthogonal array and instead use the signal-to-noise ratio as the most important evaluation criteria. The magnitudes of the differences are compared to each other to find the relatively large effects.

We selected the optimum level by giving the highest value of f_l in the experiment. For example, if $f_1 > f_2$, the optimum level is level 1 for the factor. Otherwise, level 2 is the optimum level. The two levels of seven factors listed in Table 2 were chosen based on the SNR analysis above. The optimal level of each factor is decided by the highest SNR value of either f_1 or f_2, where $f = \mathrm{A}, \mathrm{B}, \ldots, \mathrm{G}$. The optimal level is level 1 for factor A because $\mathrm{A}_1 > \mathrm{A}_2$; the optimal level is level 1 for factor B because $\mathrm{B}_1 > \mathrm{B}_2$, and so on. An orthogonal array provides an orthogonal design of the experiment matrix. When factors are assigned to the columns of an orthogonal

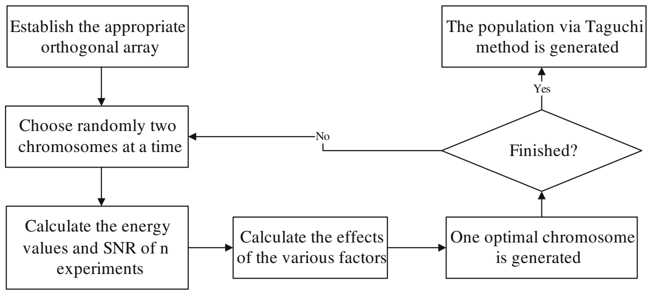
Establish the appropriate orthogonal array
Choose randomly two chromosomes at a time
Calculate the energy values and SNR of n experiments
Calculate the effects of the various factors
One optimal chromosome is generated
Finished?
No
Yes
The population via Taguchi method is generated

Fig. 4. The flowchart of select optimal factors via Taguchi method.

Table 2
Effect table for SNR on selection of optimum levels.

Experiment number (i)	Factors (f)								
	A	B	C	D	E	F	G	E^*	η_i
1	1	1	1	1	1	1	1	−4	16
2	1	1	1	2	2	2	2	−2	4
3	1	2	2	1	1	2	2	−2	4
4	1	2	2	2	2	1	1	−3	9
5	2	1	2	1	2	1	2	−2	4
6	2	1	2	2	1	2	1	−4	16
7	2	2	1	1	2	2	1	−2	4
8	2	2	1	2	1	1	2	−2	4
f_1	33	40	28	28	40	33	45		
f_2	28	21	33	33	21	28	16		
Optimal level	1	1	2	2	1	1	1		

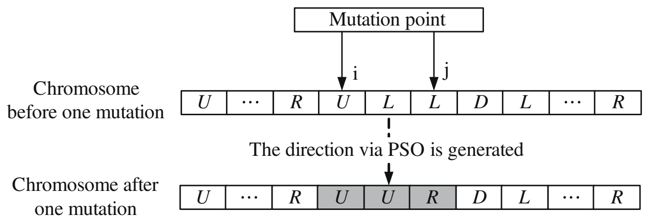
Mutation point
i
j
Chromosome before one mutation
U ⋯ R U L L D L ⋯ R
The direction via PSO is generated
Chromosome after one mutation
U ⋯ R U U R D L ⋯ R

Fig. 5. Mutation operation using PSO with ith–jth chromosomes.

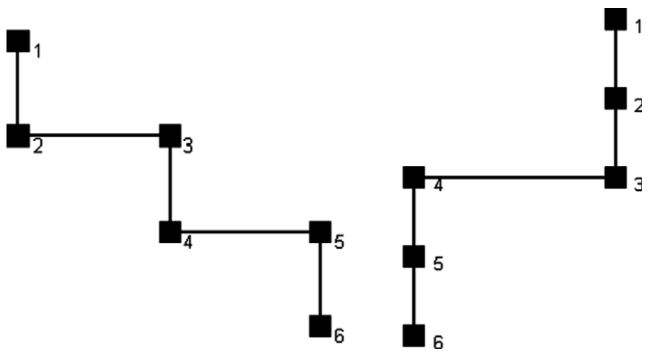

Fig. 7. Rotation motion.

Table 3
The residue folds direction with local search.

Original direction	Opposite direction	CW direction	CCW direction
Right (*R*)	*L*	*D*	*U*
Left (*L*)	*R*	*U*	*D*
Up (*U*)	*D*	*R*	*L*
Down (*D*)	*U*	*L*	*R*

Table 4
The 2D HP benchmarks.

Sequence	Length	Protein sequence	Energy
1	20	$(HP)_2PH(HP)_2(PH)_2HP(PH)_2$	−9
2	24	$H_2P_2(HP_2)_6H_2$	−9
3	25	$P_2HP_2(H_2P_4)_3H_2$	−8
4	36	$P(P_2H_2)_2P_5H_5(H_2P_2)_2P_2H(HP_2)_2$	−14
5	48	$P_2H(P_2H_2)_2P_5H_{10}P_6(H_2P_2)_2HP_2H_5$	−23
6	50	$H_2(PH)_3PH_4PH(P_3H)_2P_4(HP_3)_2HPH_4(PH)_3PH_2$	−21
7	60	$P(PH_3)_2H_5P_3H_{10}PHP_3H_{12}P_4H_6PH_2PHP$	−36
8	64	$H_{12}(PH)_2((P_2H_2)_2P_2H)_3(PH)_2H_{11}$	−42

array, the experiment runs will be balanced. This property is very powerful compared to a traditional shotgun approach or a one-factor-at-a-time approach. When an L_8 is used, there will be 8 combinations of factors or 8 experimental runs. In other words, all directions of the protein folding pathway will be tried out for factor collection. Finally, the population via Taguchi method is generated. We have updated the bottom-half of the population.

3.3.5. Mutation step

When the mutation points are selected, the mutation of an individual is as shown in Fig. 5. For the HP protein folding problem, every amino-acid residue owns each folding direction that is unique to the site. GA is incapable of efficient mutation operations aimed at each residue. Therefore, we propose a mutation operation based on particle swarm optimization.

Eq. (4) represents variations of the folding direction to the next generation. We calculate the difference between the current particle and the local best particle. We also calculate the difference between the current particle and the global best particle. Eq. (5) represents variations of the current position that updates the particles. Through the mutation step, only one best child can survive to replace its parent and enter the next generation. This involves PSO changing the local structure between i and j, where i and j are two randomly determined sequence positions, such that $1 \leqslant i \leqslant j \leqslant n$ (the length of protein sequence). Hence, we employ the merits of PSO to improve mutation mechanism.

In order to avoid trapping in a local optimal solution and to ensure the searching capability of the near global optimal solution, mutation in PSO plays an important role. It is a recently invented high performance optimizer that possesses several highly desirable attributes, including the fact that the basic algorithm is very easy to understand and implement. It requires less computational memory and fewer lines of code. Each particle has a velocity vector v and a position vector x to represent a possible solution. Each particle has three choices in evolution: (1) Insists on itself. (2) Moves towards the optimum at present. Each particle remembers its own

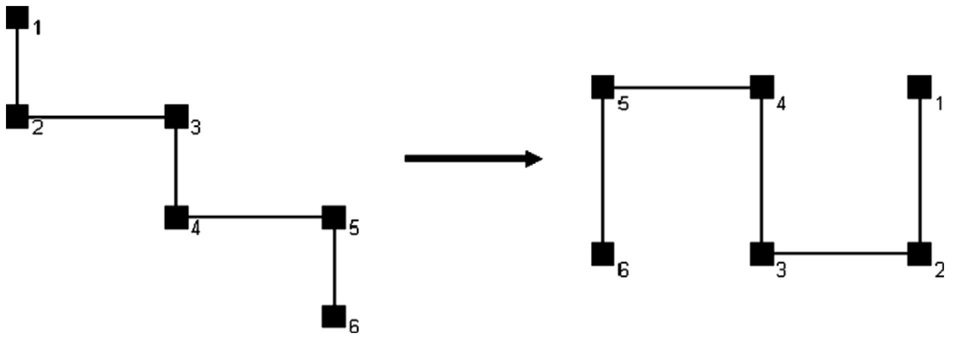

Fig. 6. Opposite motion.

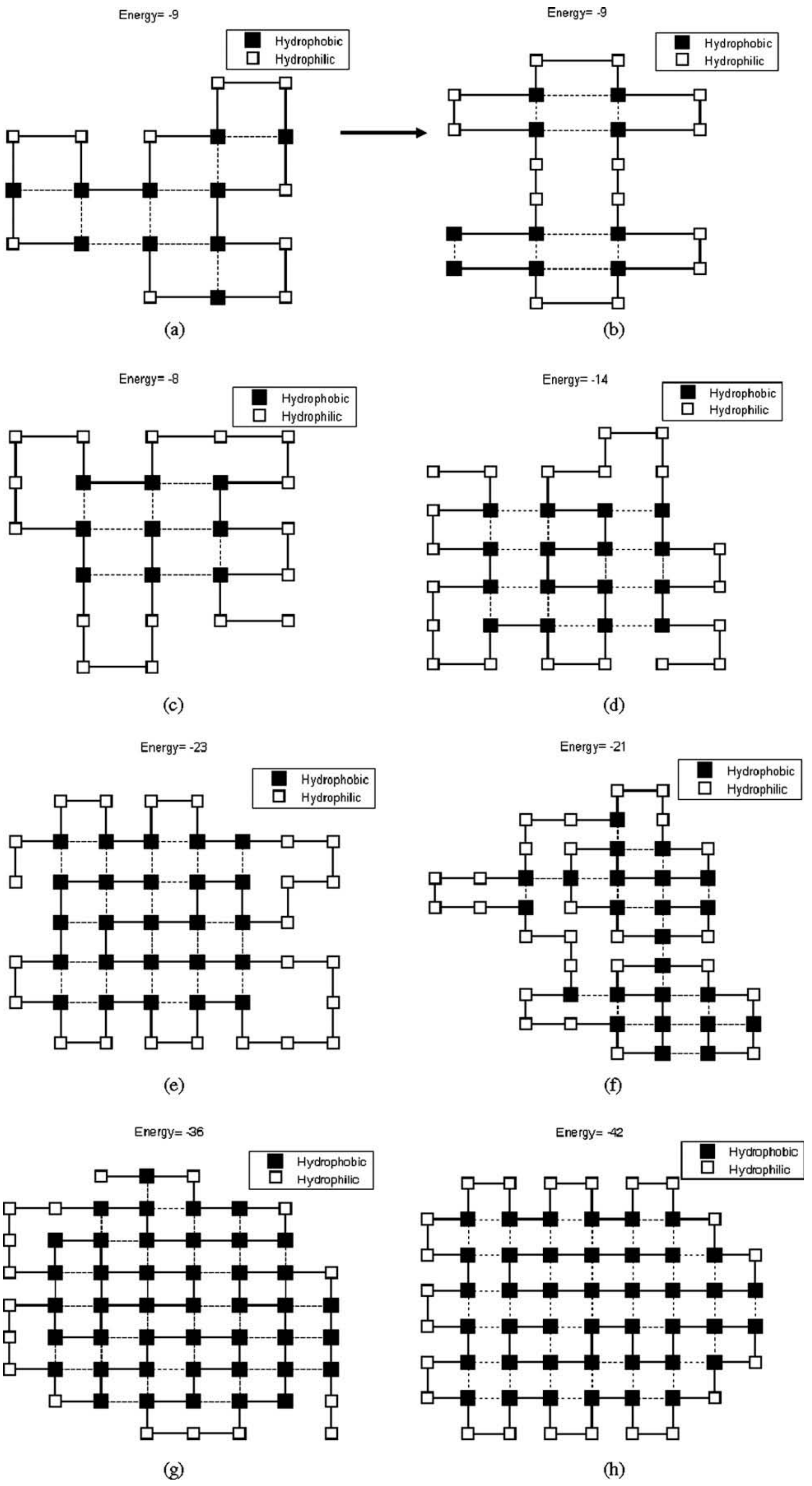

Fig. 8. Results of the structure of 8 protein sequences.

Table 5
Comparison of our approach with the genetic algorithm (GA), ant colony optimization (ACO), Monte Carlo (MC), and tabu search with genetic algorithm (GTS).

Sequence	Length	E^*	Our methods		GA (Unger & Moult, 1993)	ACO (Shmygelska et al., 2002)	MC (Liang & Wong, 2001)	GTS (Jiang et al., 2003)
			Taguchi + GA + PSO	GA + PSO				
1	20	−9	−9	−9	−9	−9	−8	−9
2	24	−9	−9	−9	−9	−9	−8	−9
3	25	−8	−8	−8	−8	−8	−7	−8
4	36	−14	−14	−14	−12	−14	−12	−14
5	48	−23	−23	−23	−22	−23	−18	−23
6	50	−21	−21	−21	−21	−21	−19	−21
7	60	−36	**−36**	−35	−34	−34	−31	−35
8	64	−42	**−42**	−38	−37	−32	−31	−39

Table 6
Performance of HTGA on the benchmark sequence.

Number	1	2	3	4	5	6	7	8
Size	20	24	25	36	48	50	60	64
E^*	−9	−9	−8	−14	−23	−21	−36	−42
Mean	−9	−9	−7.42	−13.8	−22.44	−20.52	−33.66	−38.56
SD	0	0	0.532	1.00	1.899	1.628	1.716	2.316

personal best position that it has found, called the local best. (3) Moves towards the best the population has met. Each particle also knows the best position found by any particle in the swarm, called the global best.

3.3.6. Local search step

After mutation, a local search is the same as the crossover operation. New individuals are created by changing the site's direction between point i and point j. Two motions have produced a new direction. The new folding direction is superior to the original direction. One of the two motions is selected by competition. If the new folding direction is not better than the original direction, the original direction will not change.

A local search can perform an intensive search for a new and better solution. This is similar to the mutation operation. The local search is different from the mutation operation in terms of the rules. A local search has system rules and effectively finds a local solution.

3.3.7. Opposite motion

As shown in Fig. 6, the motion of a local structure is in the opposite direction. We can change the local structure between two randomly determined sequence positions. The second residue to the fifth residue directions are changed from *right, down*, and *right* to *left, up*, and *left*.

3.3.8. Rotation motion

The rotation motion is divided into rotation clockwise (CW) and counterclockwise (CCW). As shown in Fig. 7, the second to the fifth residue directions are changed from *right, down*, and *right* to *down, left*, and *down*.

Therefore, Table 3 shows the relationship between the original directions and the transformed directions. We choose the best direction in a local search by three methods. The new folding direction is superior to the original direction. If the new folding direction is not better than the original direction, the original direction will be not changed.

3.3.9. Updating the local best and the global best

In this step, we update the local best and the global best. If the fitness value of a particle is higher than that of the local best, then the local best will be replaced with the particle; and if the local best is better than the current global best, than we replace the local best with the global best in the swarm.

3.3.10. Termination condition

The algorithm is run for a maximum of 2000 iterations or until minimum free energy is achieved in the sequence. The best member of the population is then returned.

4. Simulation results

In this section, we compared our method with the traditional genetic algorithm (Unger & Moult, 1993), the ant colony algorithm (Shmygelska, Anguirre-Hernandez, & Hoos, 2002), Monte Carlo (Liang & Wong, 2001), and the tabu search with the genetic algorithm (Jiang et al., 2003). In Table 4, the 8 chosen HP instances are standard benchmarks used to test the searching ability of the algorithms. The free energy is the optimal or best-known energy value. $H_i, P_i e(\dots)_i$ indicates i repetitions of the relative symbol or subsequence. Sequences 1 through 8 were introduced in Jiang et al. (2003). These sequences have been used as the benchmark for the 2D HP model.

We give the structure obtained by our algorithm as follows. Fifty independent runs of the algorithms were performed. For sequences 1 through 3, a population size of 100 was used. For sequences 4 through 6, a population size of 200 was used. Sequences 7 and 8 used a population size of 300. For all sequences, 2000 iterations of our algorithm were run. The crossover rate and mutation rate were set to 0.8 and 0.1, respectively. We also chose $c_1 = c_2 = 1$ in Eq. (4). These sets of parameters were experimentally determined. The structure of 8 protein sequences can be clearly seen in Fig. 8a–h.

Table 5 shows a performance comparison of various existing algorithms. GA was implemented in (Unger & Moult, 1993). For all sequences, 300 iterations of GA were run with a population size of 200. The state-of-art algorithm for the 2D HP problem is an ant colony optimization (ACO) algorithm (Shmygelska et al., 2002). The ACO was run with 10,000 iterations. The results of Metropolis Monte Carlo are reported in Liang and Wong (2001). Each run of MC consisted of 50,000,000 iterations. For sequence 1 through 3 and 6, the tabu search with genetic algorithm (GTS) (Jiang et al., 2003) was run for 100 iterations with a population size of 400. For sequence 4, 100 iterations were run with a population size of 300. For sequences 5, 7, and 8, 100 iterations were run with a population size of 1000. GTS was executed with fewer iterations than others, but the tabu search was time-consuming.

In Table 5, we also tested the traditional GA combined with particle swarm optimization with this problem. Though the results are worse than GTS, it was better than the traditional GA. The results indicate that our approach performs very well against existing evolutionary algorithms. Our approach has obvious advantages over other evolutionary algorithm for long sequences. In particular, we found the free energy for protein sequence 7 to be energy = -36, and, for sequence 8, energy = -42. The results are summarized in Table 6 in terms of the best found free energy (E^*), mean, and standard deviation (SD). It is important to note that the standard deviation keeps a small value when the length of a sequence increases.

5. Conclusion

This paper proposed an efficient hybrid Taguchi-genetic algorithm for solving protein structure prediction problem. The proposed hybrid algorithm combines the genetic algorithm, Taguchi method, Particle Swarm Optimization, and a local search of the protein folding pathway. The Taguchi method was used obtain the optimum offspring, while the particle swarm optimization was used to improve mutation operation of the genetic algorithm. In mutation operation based on particle swarm optimization, the cognitive component encourages the particles to move toward their own best positions. We demonstrated that our algorithm can be applied successfully to the protein folding problem based on the 2D hydrophobic–hydrophilic lattice model. Simulation results indicate that our approach performs very well against existing evolutionary algorithms. In future work, we intend to tackle the prediction of 3D structures for protein folding using the proposed algorithm and to test other local search strategies to enhance the predictive ability.

References

Bui, T. N., & Sundarraj, G. (2005). An efficient genetic algorithm for predicting protein tertiary structures in the 2D HP model. In *Proceedings of the 2005 conference on Genetic and evolutionary computation (GECCO'05)* (pp. 385–392).

Cordon, O., Herrera, F., Hoffmann, F., & Magdalena, L. (2001). *Genetic fuzzy systems evolutionary tuning and learning of fuzzy knowledge bases. Advances in fuzzy systems-applications and theory* (Vol. 19). NJ: World Scientific Publishing.

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

Eberhart, R., & Kennedy, J. (1995a). A new optimizer using particle swarm theory. In *Proceedings of the sixth international symposium on Micro Machine and Human Science (MHS'95)* (pp. 39–43).

Eberhart, R., & Kennedy, J. (1995b). Particle swarm optimization. *IEEE International Conference on Neural Networks, 4*, 1942–1948.

Guo, Y. Z., Feng, E. M., & Wang, Y. (2007). Optimal HP configurations of proteins by combining local search with elastic net algorithm. *Journal of Biochemical and Biophysical Methods, 70*(3), 335–340.

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

Kaytakoğlu, S., & Akyalçın, L. (2007). Optimization of parametric performance of a PEMFC. *International Journal of Hydrogen Energy, 32*(17), 4418–4423.

König, R., & Dandekar, T. (1999). Refined genetic algorithm simulation to model Proteins. *Journal of Molecular Modeling, 5*(12), 317–324.

Krasnogor, N., Pelta, D., Lopez, P. M., Mocciola, P., & de la Canal, E. (1998). Genetic algorithms for the protein folding problem: A critical view. In C. F. E. Alpaydin (Ed.), *Proceedings of engineering of intelligent systems*. ICSC Academic Press.

Krasnogor, N., Hart, W. E., Smith, J., & Pelta, D. A. (1999). Protein structure prediction with evolutionary algorithms. *Proceedings of the genetic and evolutionary computation conference* (pp. 1596–1601). Orlando, FL, Morgan Kaufmann, USA.

Liang, F. M., & Wong, W. H. (2001). Evolutionary Monte Carlo for protein folding simulations. *Journal of Chemical Physics, 115*(7), 3374–3380.

Lin, C. J., & Hong, S. J. (2007). The design of neuro-fuzzy networks using particle swarm optimization and recursive singular value decomposition. *Neurocomputing, 71*, 297–310.

Lin, C. J., & Hsu, Y. C. (2007). Reinforcement hybrid evolutionary learning for recurrent wavelet-based neuro-fuzzy systems. *IEEE Transactions on Fuzzy Systems, 15*(4), 729–745.

Lin, C. J., Liu, Y. C., & Lee, C. Y. (2008). An efficient neural fuzzy network based on immune particle swarm optimization for prediction and control applications. *International Journal of Innovative Computing Information and Control (IJICIC), 4*(7), 1711–1721.

Liu, Y. T., Fung, R. F., & Wang, C. C. (2007). Application of the nonlinear, double-dynamic Taguchi method to the precision positioning device using combined piezo-VAM actuator. *IEEE Transactions on Ultrasonics, Ferroelectronics, and Frequency Control, 54*(2), 240–250.

Patton, A. L., Punch, W. F., III, & Goodman, E. D. (1995). A standard GA approach to native protein conformation prediction. In *Proceedings of the sixth international conference genetic algorithms* (pp. 574–581). Morgan Kauffman.

Pedersen, J., & Moukt, J. (1997). Protein folding simulations with genetic algorithms and a Detailed Molecular Description. *Journal of Molecular Biology, 269*(2), 240–259.

Ross, P. J. (1989). *Taguchi techniques for quality engineering*. New York: McGraw-Hill.

Roy, R. K. (1990). *A primer on the Taguchi method*. New York: Van Nostrand Reinhold.

Shmygelska, A., Anguirre-Hernandez, R., & Hoos, H. H. (2002). An ant colony optimization algorithm for the 2D HP protein folding problem. In Proceedings of the international workshop ant algorithms (pp. 40–52). Brussels, Belgium.

Taguchi, G., Chowdhury, S., & Taguchi, S. (2000). *Robust engineering*. New York: McGraw-Hill.

Takahashi, O., Kita, H., & Kobayashi, S. (1999). Protein folding by a hierarchical genetic algorithm. In *Proceedings of the fourth international symposium on Artificial Life and Robotics* (pp. 334–339). Oita, Japan.

Unger, R., & Moult, J. (1993). Genetic algorithms for protein folding simulations. *Journal of Molecular Biology, 231*(1), 75–81.

Wang, T. Y., & Huang, C. Y. (2008). Optimizing back-propagation networks via a calibrated heuristic algorithm with an orthogonal array. *Expert Systems with Applications, 34*(3), 1630–1641.

Wu, Y., & Wu, A. (2000). *Taguchi methods for robust design*. New York: ASME.

Yap, A., Cosic, I. (1999). Application of genetic algorithm for predicting tertiary structure of peptide chains. In *Proceedings of the first joint BMES/EMBS conference on serving humanity, advancing technology* (pp. 1214). Atlanta, GA, USA.