An Adaptive Discrete Brain Storm Algorithm Solves 3D Protein Structure Prediction

Alaa Fahim[1]
Math Department, Faculty of science,
Assuit University,
Assuit, Egypt

Nehad Abdelraheem[2]
Faculty of Computer Science and Information System,
Assuit University,
Assuit, Egypt

Abstract—**Brain Storm Optimization (BSO) is one of the major effective swarm intelligence algorithms that simulate the human brainstorming process to find optimality for optimization problems. BSO method has successfully been applied to many real-world problems. This study employs BSO method, called BSO-IP, to solve the integer programming problem. Our method collects best solutions to generate new solutions that then search for optimal solutions in all areas of search space.The BSO-IP method solves some benchmark integer programming problems to test its efficiency. The BSO-IP is used to simulate the 3D protein structure prediction problem, which is mathematically presented as an integer programming problem to approve the viability and helpfulness of our proposed Algorithm. The experimental results of different benchmarks protein structure show that our proposed method is superior in high performance, convergence, and stability in predicting protein structure. We examined our strategy results to be promising compared to other results.**

Keywords—***Brain storm optimization; integer programming problem; three dimensional protein structure prediction***

I. Introduction

The optimization problem is a significant branch of modern science problem. Previously, Scientists took more time to find an optimal solution to these problems. However, recently, widely researched Optimization problems depend on the population. The algorithms for this subject are called population-based optimization algorithm. The population- based optimization problem works by communicating and competing with each other, and its optimization algorithms are classified as swarm intelligence algorithms.

Particle swarm optimization (PSO) [1], bacterial foraging optimization [2] , artificial bee colony optimization [3], and ant colony optimization (ACO) [4] are examples of PSO are inspired by animals and insects such as ants, birds, and bees. Brain Storm Optimization (BSO) is a new type of PSO, proposed by Shi [5, 6]. Many researchers play significant efforts to develop the BSO algorithm to make it more efficient.

BSO depends on two major functions, namely, divergence and convergence. Learning and developing capabilities are the two basic functions that BSO possesses. Divergence correlates with learning and convergence with developing capabilities. These functions find better possible solutions than the current solution, which depends on one member of the population. These two functions are essential to finding the best potential solutions to solve (NP) problems. The BSO algorithm is a mixture of swarm intelligence and data mining techniques. Each solution produced using the BSO algorithm not only solves the problem but is also an outlet to other solutions to the problem. This feature is the sole characteristic of combining swarm intelligence and data mining techniques.

Most of BSO algorithms are employed to solve the continuous optimization problem [7] and [8]. Only a few papers have been dedicated integer programming problems and their real applications like [9]. This study employs the BSO algorithm ,called the BSO-IP, to solve an integer programming problem, to solve some benchmark integer programming problems, Also, BSO-IP results were compared with those from other methods to show our method strength. The BSO-IP method makes an adaptive update to solutions by collects the best solutions to help to generate new solutions that differentiate them to search for optimal solutions in all areas of the search space.

This paper presents the BSO Algorithm approach to one of the most important problems in bioinformatics, which is the protein structure prediction (PSP) in 3D. PSP is characterized by forecasting of the 3D structure of a protein using its essential structure data. PSP is a significant research topic in bioinformatics, medication, and different fields such as sedate structure, and the forecast of maladies. The dimensional folding structure of a protein determines its biological function. There are many traditional experimental methods to determine protein folding structure such as X-ray crystallography and NMR spectroscopy [10].

PSP is presented as a mathematical form, which is an integer programming. BSO-HP algorithm, simulated to solve different benchmarks benchmarks HP model. is used to test the effectiveness of the BSo-HP algorithm.

However, they are very expensive and time-consuming because of the polypeptide chain structures such enormous number of various spatial structures. It is as yet difficult to look for the global minimum energy conformations of proteins from its sequence of amino acids and make analysis for the protein folding process. The most series problem lies in finding the simplest model representing the relationship between the structure of a protein and free energy.

Whatever is left of the paper is sorted out as taken after. In Section II, we highlight the fundamental techniques and structure for the BSO method and briefly review of the integer programming problem. The design of the proposed methods for the integer programming problem known as BSO-IP is introduced, and the numerical experiments of the BSO-IP method are discussed in Section III. The BSO method is applied to solve PSP as HP- BSO strategy in Section IV. Furthermore the correlation between the proposed technique and different strategies in Section V, Finally, Section VI shows the conclusions of this paper.

II. Related Work

A. Brain Storm Optimization Method Techniques

The BSO algorithm was designed by Shi [5, 6] like other swarm intelligence optimization algorithm but inspired by the brain of the human brain processing. Humans are the smartest living creatures ever,so algorithms based on humans and on human behavior are more effective and rewarding than those from insects, ants, and other living things.

The BSO algorithm is designed according to the brain-storming process. Osborn created four rules to generate the idea. Open-minded people generate many different ideas during brainstorming. Every population in the BSO algorithm contains a group of diverse ideas. At the end of every step of brainstorming, Every population in the BSO algorithm contain a group of diverse ideas. At the end of every step of brainstorming , every idea will be evaluated. Therefore, no idea ignored.

There are five major operations of the BSO algorithm are shown in Fig. 1 with the following description:

- Population initialization.
- Evaluating individuals.
- Clustering individuals.
- Disrupting cluster centers.
- Updating individuals.

In initialization, populations are generated randomly from the normal distribution inside the search space, and the size of the population is constant at every iteration. It is necessary to evaluate each individual after each generation because evaluated value determines the competence of the individual as the potential solution. Many of clustering types can be used in the clustering step however, the K-means clustering algorithm is applied in the BSO algorithm. The updating individual step includes two suboperation presented in the following equations:

$$\begin{aligned} x^i_{new} &= x^i_{old} + \zeta(t) + random(t) \\ x^i_{old} &= \omega 1 * x^i_{old1} + \omega 2 * x^i_{old2} \end{aligned} \quad (1)$$

Where x^i_{old} is the summation of i-dimensional of x^i_{old1} and x^i_{old2} weights, and ω_1 *and* ω_2 are coefficients for weighting two existing individuals. ω_1 and ω_2 2 equal 0 if new individual x^i_{new} is generated depending on existing individual x^i_{old}. And if it depends on two existing individuals $x^i_{old1}$and x^i_{old2} , then the coefficient ζ(t) is randomly generated by one possibly function:

$$\zeta(t) = \log sig \left| \frac{\frac{T}{2} - t}{k} \right| * random(t) \quad (2)$$

where logsig() is a logarithmic sigmoid transfer function, T is the maximum number of iterations, t is the current iteration number, k is the chang in slope of the logsig(), and random() is a random value within (0,1).

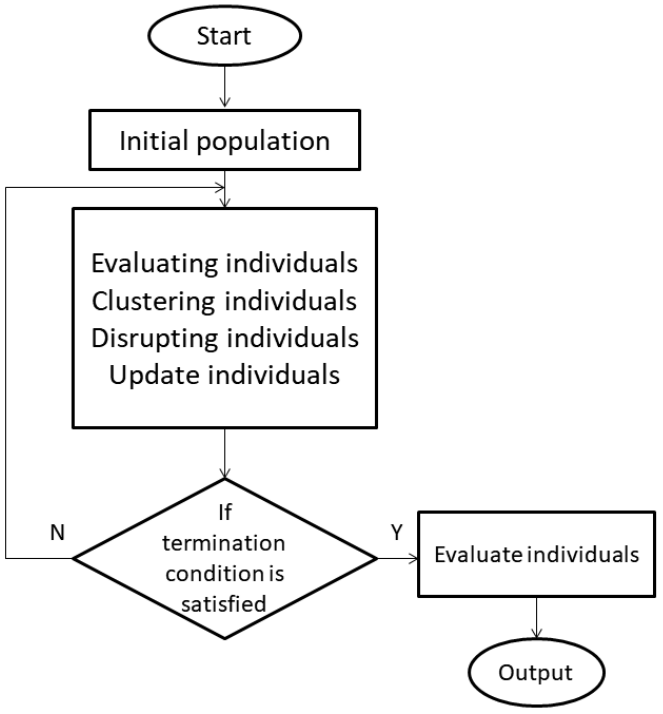


Fig. 1. BrainStorm Optimization Flowchart.

B. Integer Programming Problem

An integer programming problem, as an optimization problem in mathematical form, contains a few or the entirety of the variables confined to be integers. The mathematical programming problem can be represented in a mathematical form as follows:

$$\begin{aligned} \min \quad & f(\mathbf{y}), & \\ s.t. \quad & g_i(\mathbf{y}) < 0, & i = 1, \ldots, I, \\ & h_j(\mathbf{y}) = 0, & j = 1, \ldots, J, \\ & \mathcal{L}_l \le y_l \le \mathcal{U}_l, & l = 1, \ldots, n, \end{aligned} \quad (3)$$

where f, g, h are nonconvex functions in the general case, and n is the number of discrete variables. $\mathcal{L}_{\mathbf{y}} = (\mathcal{L}_1, \ldots, \mathcal{L}_n)$, and $\mathcal{U}_{\mathbf{y}} = (\mathcal{U}_1, \ldots, \mathcal{U}_n)$, are thelower and upper bounds for discrete variables, respectively. Problem (3) is the same as general nonlinear programming except that the design variables can take on any form of zero-one, integer and discrete variables. Therefore, the penalty methodology [11] was employed to transform this constrained problem into series of unconstrained problems, whose unconstrained solutions converge to the solutions of the constrained problem.

1) Penalty Function: The penalty method transforms a constrained optimization problem by a series of unconstrained problems whose solutions must converge to the solution of the original constrained problem. In the case of minimization with inequality constraints, the corresponding minimization problems are formed by adding a penalty term to the objective function. The penalty term grows when the constraints are violated and is set to zero in the region where constraints are not violated. The penalty term is usually a product of a positive

penalty coefficient and a penalty function.
We try to solve the constrained Equation (3), where f , g_i and h_i are real valued function defined in search space, $S \subset R^n$ The general formulation of the exterior penalty function is:

$$\varphi(y) \; - f(y) \pm [\sum_{i=1}^{I} \; r_i + G_i + \sum_{j=1}^{j} \; c_j \; \times L_j] \qquad (4)$$

Where $\varphi(y)$ is the new objective, G_i and L_i are called constraint violation function, and most common form is for them are

$$\begin{aligned} G_i &= max[0, g_i(y)]^\alpha \\ L_j &= [h_j(y)]^\beta \end{aligned} \qquad (5)$$

Where $\alpha \; and \; \beta$ are normally 1 or 2. There are many different formulations of penalty function.

III. An Adaptive Discrete Brain Storm Optimization Algorithm for Integer Programming Problem

A discrete BSO method is simulated to solve integer programming problem as NP problem, which is called BSO-IP. The key operations of the BSO-IP algorithm are designed as the following description.

A. *Initial Population*

Initial population p was created from the uniform random distribution inside the search space. The population size pop_num is a fixed around all search processes.

B. *Clustering Individuals and Disrupting Cluster Centers*

Clustering analysis is considered unsupervised learning. It is a technique to divide data into several groups. The m goal of clustering algorithms is to separate data into small groups with similar and related objects. There are two ways of measuring similarity in the clustering analysis: first, finding an intercept between objects. In another way, the distance between the objects is calculated or measured; second, calculating distance is the common way to measure the similarity in clustering. The clustering process is similar to the brainstorming process of dividing ideas into small groups with similar objects. We applied the K-mean clustering algorithm [12] because its efficiency and accurate computation. Procedures 3.1 demonstrates the clustering technique.

Procedure 3.1: Clustering Technique
1. Let X = $x_1, x_2, \ldots, x_n$ be the set of data points and V = $v_1, v_2, \ldots, v_c$ be the set of centers.
2. Randomly select 'c' cluster centers
3. Calculate the distance between each data point and cluster centers using k-mean algorithm.
4. Assign the data point to the cluster center whose distance from the cluster center is the minimum of all cluster centers.
5. Recalculate the new cluster center using: where, 'c_i' represents the number of data points in the ith cluster.
6.Recalculate the distance between each data point and newly obtained cluster centers using the k-mean algorithm.
7. If no data point was reassigned then stop, otherwise repeat from Step 3.

C. *Updating Individuals*

1) New Individual Generation: To generate new individuals, we employ the prior information the best individuals saved in *Best-list*. The best individuals are generated after generating the initial population.

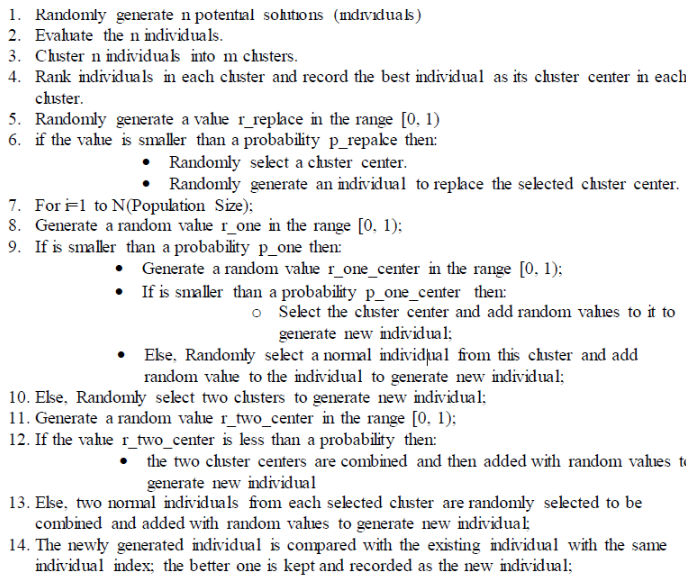
A new individual is generating based on one or two clustering centers with the following details:

- One Individual Mutation
  Procedure 3.2: One Individual Mutation
  1. Select one clustering center randomly P_C.
  2. Generate r_mutate as a random number in $[0, 1]$
  3. If $r_mutate < 0.6$ do Step 4, otherwise do Step 5.
  4. Select any gene in P_C and change its value from the individual in *Best-list*.
  5. Select two genes in P_C and change their values from the individual in *Best-list*.

- Two Individuals Operators
  Procedure 3.3: two Individuals operators
  1. Select two clustering centers randomly P_{C1} and P_{C2}.
  2. Determine the similar part in P_{C1} with the best individuals in *Best-list*.
  3. Change the selected part in P_{C1} with the corresponding part in P_{C2}.

2) Selection: When executing the algorithm, the population size does not change; rather, it is fixed. In each iteration, a new individual is replaced with the old individual. The replacement follows the selection technique: preserving the best by comparing the new individual to the old individual in the same index and choosing the best. Finally, the *Best-list* is updated with the enhancement individuals.

The BSO-IP algorithm criteria are presented in Fig. 2.



```
1.  Randomly generate n potential solutions (individuals)
2.  Evaluate the n individuals.
3.  Cluster n individuals into m clusters.
4.  Rank individuals in each cluster and record the best individual as its cluster center in each
    cluster.
5.  Randomly generate a value r_replace in the range [0, 1)
6.  if the value is smaller than a probability p_repalce then:
        • Randomly select a cluster center.
        • Randomly generate an individual to replace the selected cluster center.
7.  For i=1 to N(Population Size);
8.  Generate a random value r_one in the range [0, 1);
9.  If is smaller than a probability p_one then:
        • Generate a random value r_one_center in the range [0, 1);
        • If is smaller than a probability p_one_center then:
              o Select the cluster center and add random values to it to
                generate new individual;
        • Else, Randomly select a normal individual from this cluster and add
          random value to the individual to generate new individual;
10. Else, Randomly select two clusters to generate new individual;
11. Generate a random value r_two_center in the range [0, 1);
12. If the value r_two_center is less than a probability then:
        • the two cluster centers are combined and then added with random values t
          generate new individual
13. Else, two normal individuals from each selected cluster are randomly selected to be
    combined and added with random values to generate new individual;
14. The newly generated individual is compared with the existing individual with the same
    individual index; the better one is kept and recorded as the new individual;
```

Fig. 2. Brain Storm Optimization Algorithm.

D. Numerical Experiment

The values of some parameters are set to the values reported in the literature. Other parameters are set with a preliminary numerical experiment. The values of parameters are presented in Table I.

TABLE I. BSO-IP PARAMETERS

Parameter Operator	Paramter Value	Description
cluster num	5	The number of k-means clusters
p_replace	0.4	The probability of replacing operator
p_one	0.4	The probability of selecting one cluster
p_one_center	0.3	The probability of selecting the center of one cluster
p_two_center	0.2	The probability of selecting The centers of two clusters

1) Results of Unconstrained Problems: The BSO-IP method is applied to solve eight unconstrained well-known problems which are showed in Table II.

TABLE II. UNCONSTRAINED FUNCTIONS

functions	Definition	Range	optimal solution
g_1	$\|y_1\| + \|y_2\| + \ldots + \|y_D\|$	$D = 5, 10, 15, 20, 25, 30$	0
g_2	$Y^T Y$	$D = 5$	0
g_3	$(9y_1^2 + 2y_2^2 - 11)^2 + (3y_1 + 4y_2^2 - 7)^2$	$D = 2$	0
g_4	$(y_1 + 10y_2)^2 + 5(y_3 - y_4)^2 + (y_2 - 2y_3)^4 + 10(y_1 - y_4)^4$	$D = 4$	0
g_5	$2y_1^2 + 3y_2^2 + 4y_1y_2 - 6y_1 - 3y_2$	$D = 2$	-6
g_6	$-3804.84 - 138.08y_1 - 232.92y_2 + 123.08y_1^2 + 203.64y_2^2 + 182.25y_1y_2$	$D = 2$	-3833.12
g_7	$(y_1 - 2)^4 + (y_1 - 2y_2)^2$	$D = 2$	0
g_8	$y_1^2 - 4y_1 - 2y_1y_2 + 2y_2^2$	$D = 2$	-8

g_1 has various dimensions n=5, 10, 15, 20, 25, 30. The problems from g_1 to g_6 are mentioned in [13],whereas g_7 and g_8 problems are mentioned in [14].The BSO-IP method is programmed in MATLAB and ran 50 times to get the results, which satisfy the termination condition of obtaining the optimal solution with errors 10^{-3} or to get the maximum number of iterations.

Table III presents results for the BSO-IP method. Also, g* is the known solution, g-best is the best solution obtained by the proposed method, g-mean is the mean of the optimal values, SR is the success rate, and g-evolution is the fitness function evolution.

TABLE III. BSO-IP METHOD FOR UNCONSTRAINED PROBLEMS

g	n	g*	g-best	g-mean	SR	g-evolution
1	5	0	0	0	100	114.4
1	10	0	0	0	100	113.8
1	15	0	0	0	100	114.8
1	20	0	0	0	100	112.5
1	25	0	0	0	100	117.4
1	30	0	0	0	100	118.6
2	5	0	0	0	100	112.4
3	2	0	0	0	100	77.62
4	4	0	0	0	100	114.26
5	2	-6	-6	-6	100	62.42
6	2	-3833.12	-3833.12	3833.12	100	135.2
7	2	0	0	0	100	113
8	2	8	8	8	100	2840

We compare the BSO-IP method with PSO-In, PSO-Co, PSO-BO, and BB methods [15]. The BSO-IP Algorithm ran 30 times under termination conditions to reach to the exact solution with accuracy 10^{-6} or got 2500 as presented in PSO-In, PSO-Co, PSO-BO, and BB methods in Table IV. Alternatively, Table V presents the comparison between our proposed method and PSO-In, PSO-Co, PSO-BO,and BB methods. The results show that the BSO-IP method found all the optimal solutions for the test problem with the lowest fitness function evolution.

TABLE IV. COMPARISON SHOWING EXACT SOLUTION WITH ACCURACY 10^{-6} IN PSO-IN, PSO-CO, PSO-BO, AND BB METHODS

g	n	Solver	g-eval	St.D	SR	g-best
g_1	5	**BSO-IP**	**111.9**	**10.55**	**100**	**0**
		PSO-In	1646	661.5	100	0
		PSO-Co	744	86	100	0
		PSO-Bo	962.6	97	100	0
		BB	1167.38	659.8	100	0
g_2	10	**BSO-IP**	**111.9**	**10.55**	**100**	**0**
		PSO-In	1646	661.5	100	0
		PSO-Co	744	86	100	0
		PSO-Bo	962.6	97	100	0
		BB	1167.38	659.8	100	0
g_3	15	**BSO-IP**	**111.9**	**10.55**	**100**	**0**
		PSO-In	1646	661.5	100	0
		PSO-Co	744	86	100	0
		PSO-Bo	962.6	97	100	0
		BB	1167.38	659.8	100	0
g_4	20	**BSO-IP**	**111.9**	**10.55**	**100**	**0**
		PSO-In	1646	661.5	100	0
		PSO-Co	744	86	100	0
		PSO-Bo	962.6	97	100	0
		BB	1167.38	659.8	100	0
g_5	25	**BSO-IP**	**111.9**	**10.55**	**100**	**0**
		PSO-In	1646	661.5	100	0
		PSO-Co	744	86	100	0
		PSO-Bo	962.6	97	100	0
		BB	1167.38	659.8	100	0
g_6	30	**BSO-IP**	**111.9**	**10.55**	**100**	**0**
		PSO-In	1646	661.5	100	0
		PSO-Co	744	86	100	0
		PSO-Bo	962.6	97	100	0
		BB	1167.38	659.8	100	0

TABLE V. COMPARISON BETWEEN THE BSO-IP METHOD WITH PSO-IN, PSO-CO, PSO-BO, AND BB METHODS

g	n	Solver	g-eval	St.D	SR	g-best
g_2	5	**BSO-IP**	**110.3**	**19.94**	**100**	**0**
		PSO-In	1655.6	618.4	100	0
		PSO-Co	428.0	57.9	100	0
		PSO-Bo	418	83.9	100	0
		BB	139.7	102.6	100	0
g_3	2	**BSO-IP**	**75.2**	**62.2**	**100**	**0**
		PSO-In	3.4.0	101.6	100	0
		PSO-Co	297.3	50.8	100	0
		PSO-Bo	302.0	80.5	100	0
		BB	316.9	125.4	100	0
g_4	4	**BSO-IP**	**116.3**	**12.96**	**100**	**0**
		PSO-In	1728.6	518.9	100	0
		PSO-Co	1100.6	229.2	100	0
		PSO-Bo	1082.0	295.6	100	0
		BB	2754.0	1030.1	100	0
g_5	2	**BSO-IP**	**54.6**	**35.25**	**100**	**-6**
		PSO-In	178.0	41.9	100	-6
		PSO-Co	198.6	59.2	100	-6
		PSO-Bo	191.0	65.9	100	-6
		BB	211.1	15.0	100	-6
g_6	2	**BSO-IP**	**126.73**	**164**	**100**	**-3833.12**
		PSO-In	334.6	95.5	100	-3833.12
		PSO-Co	324.0	78.5	100	-3833.12
		PSO-Bo	306.6	96.7	100	-3833.12
		BB	358.6	14.7	100	-3833.12

2) Results of Constrained Problems: BSO-IP method is applied to solve constrained problems. The performance of the BSO-IP method is presented on well-known problems f_1 to f_4, [16] shown in Table VI. Our proposed method solves constrained problem by transforming it into an unconstrained problem by using the penalty function equations 5 and 4:

The BSO-IP MATLAB code runs 50 times with the termination condition is to find the exact solution with an error of 10^{-6} or to get the maximum number of iterations. The result of our method is shown in Table VII.

TABLE VI. Benchmark Constrained Functions

functions	Definition	Range	optimal solution
f_1	$y_1^2 + y_2^2 + y_3^2 + y_4^2 + y_5^2$	$D = 5$	8
f_2	$exp(-y_1) + {y_1}^2 - y_1 y_2 - 3{y_2}^2 - 6y_2 + 4y_1$	$D = 2$	-42.632
f_3	${y_1}^2 + y_1 * y_2 + 2{y_2}^2 - 6y_1 - 2y_2 - 12y_3$	$D = 3$	-68
f_4	$(y_1 + 2y_2 + 3y_3 - y_4)(2y_1 + 5y_2 + 3y_3 - 6y_4)$	$D = 4$	-6

TABLE VII. BSO-IP Method for Constrained Problem

f	n	f*	f-best	f-mean	SR	f-eval
f_1	5	8	8	8	100	151.96
f_2	2	-42.632	-42.632	-42.632	100	24.2
f_3	4	-68	-68	-68	100	312.93
f_4	4	-6	-6	-6	100	99.7

Table VIII presents the comparison between BSO-IP methods with MI-LXPM, RST2ANU and AXNUM methods [16] for 4 well-known problems f_1 to f_4. The results follow after the BSO-IP code runs 50 times and the termination condition to reach the optimal solution with an error of 0.01 or achieve the maximum number of iterations. The termination condition is presented in [16] to compare our result with other methods with the same termination condition.

Table VIII also Presents the success rate (SR), fitness evaluation f-eval, and the best solution found by the solver (f-best). The result demonstrates that the BSO-IP method is promising since it found the optimal solution with the lowest fitness function evolution.

TABLE VIII. Comparison Between BSO-IP with MI-LXPM, RST2ANU and AXNUM Methods with MI-LXPM, RST2ANU and AXNUM methods

f	Solver	f-eval	SR	f-best
f_1	**BSO-IP**	**151.96**	**100**	**8**
	MI-LXPM	171	100	8
	RST2ANU	2500	100	8
	AXNUM	863	97	8
f_2	**BSO-IP**	**24.2**	**100**	**-42.631**
	MI-LXPM	99	70	-42.631
	RST2ANU	100	35	-42.631
	AXNUM	456	91	-42.631
f_3	**BSO-IP**	**312.93**	**100**	**-68**
	MI-LXPM	10933	100	-68
	RST2ANU	1489713	2	-68
	AXNUM	45228	82	-68
f_4	**BSO-IP**	**49.7**	**100**	**-6**
	MI-LXPM	671	100	-6
	RST2ANU	2673	75	-6
	AXNUM	13820	95	-6

IV. Protein Structure Model

A. HP Lattice Model

The HP model, is such that each amino acid sequence is disconnected as an alphabetic string with H (hydrophobic amino acid) and P (hydrophilic amino acid). The protein adaptations self-keeping away from way on a 3D lattice. The primary thrust of the development of the tertiary structure is the communications among hydrophobic amino acids which are near the lattice yet not adjoining in the sequence, signified as H-H interaction. The free vitality of a protein conformation(X) is communicated by the quantity of H-H interactions. From Anfinsen's supposition [17], the arrangement structures a center in the spatial structure shield dissolvable by hydrophilic amino acids with negligible free vitality. So, the higher the H-H interactions, the lower the free vitality. We expected that the free vitality is equivalent to the smaller number of H-H interactions. HP lattice model is used to solve protein structure forecast problems on 2D and 3D lattice broadly. This study focused on the 3D HP square lattice model. Many meta-heuristics methods tried to solving HP models like genetic algorithm [18] [19] [20]. Example included memetic algorithm [21] , evolutionary strategy method [17] , ACO method [22] and the Tabu search method [23] [24]. A. Baz [21] applied a memetic algorithm to solve the 3D lattice HP model. M.T. Haque [25] used a genetic algorithm to solve the 3D HP lattice model. X. Zhang [23] presented an improved Tabu search for the 3D HP lattice model. T. Thalheim [22] applied ACO to predict PSP of HP model.

P.H.R. Gabrial [17] presented an evolutionary strategy to solve the 3D HP model. Few papers have tried to solve the PSP problem as a mathematical model [26] and [27]. We treat this problem as a simpler mathematical model than other methods; because the our mathematical model is more accurate in finding the solution and is more time efficient.

BSO algorithm solves the 3D HP model, called BSO-HP, as an integer mathematical model. The result demonstrates the strength of BSO-HP to deal with 3D HP model as NP problem.

PSP Problem as Integer Programming Problem

The following equation presents the PSP problem as an integer programming problem.

$$\begin{array}{ll} max & \sum_{a,b} f_{a,b} \\ where & f_{a,b} = \begin{cases} 1, & if\,||M_a - M_b|| = 1 \\ 0 & others. \end{cases} \end{array}$$

where $a = \{1, 2, \ldots, n-2\}$ and $b = \{a+2, \ldots, n\}$.

Three constraints describe the problem: first, the overlapping constraint, which prevents two nodes from being in the same coordinate; second, connectivity constraint, which prevents any cut or change in the protein's sequential arrangement and makes sure there exist a link to other nodes. Finally, the boundary constraint is for refusing the straight structure of the HP model.

- **Overlapping**

$$||M_i - M_j|| \geq 1$$

where $i = \{1, 2, \ldots, n-1\}$ and $j = \{i+1, \ldots, n\}$

- **Connectivity**

$$||M_i - M_{i+1}|| = 1$$

where $i = \{1, 2, ..., n-1\}$.

- **Bounding**

$$\begin{array}{l} length(X) < \ graphboundary \\ length(Y) < \ graphboundary \\ length(Z) < \ graphboundary \end{array}$$

Where *graphboundary*=n/3; and *length(X)*, *length(Y)* and *length(Z)* are the length of the HP model in all three directions, respectively.

B. BSO-HP Algorithm

BSO-HP algorithm solves the PSP problem on the basis of biological theory. Thus, the BSO-HP algorithm has been applied to all procedures in the BSO-IP algorithm besides some procedures to deal with the 3D HP protein structure.

First, protein structure used the following description to write the individual on the BSO-HP algorithm:

- Protein sequence can be written as the chain of amino acid donated as S vector, $S = \{s_1, \ldots, s_n\}$ where n donates the length of the protein sequence, Each s in the S vector may be H or P monomers.

- Denote the direction by vector; X: it contains the direction of each three monomers, X vector has a length of n-2, and each direction is in the range of 0 to 4, where 0 means forward, 1 means left, 2 means right, 3 means up and 4 means down.

- Finally, matrix M involves the coordinate of each node (x, y, z). The nodes in beginning take two coordinates (0,0,0) and (0,0,1).

TABLE IX. Coordinate Nodes in HP Model

M_x	M_y	M_z
0	0	0
0	0	1
1	1	0
1	2	0
1	3	0
0	3	0
0	4	0
0	5	0
0	6	0
-1	6	0
-1	5	0
-1	4	0
-1	3	0
-1	2	0
-2	2	0
-2	2	-1
-2	2	-2

Fig. 3 shows HP model with white nodes for P monomers and black for H monomers. Applying the previous description of the protein structure, then the first S= {HHHHPPHHHHHHHPPPH}, with direction vector X= {2, 4, 4, 2, 1, 0, 3, 2, 2, 4, 3, 3, 1, 3, 2, 3, 2} and coordinate nodes in M matrix is presented in Table IX : From the structure of the protein, we will implement some procedures like initial population and updating individual procedures from the BSO-IP and used in the BSO-HP algorithm are well described below.

1) Initial Population: Every individual is represented by the direction of two nodes generated random values of length n-1, where n is the length of a sequence of protein lattice. For example , X={0, 1, 2, 3, 1, 4, 2,...., 4}. Procedure 4.1 will introduce how we generate the initial solution:

Procedure 4.1: Initial Solution
1. The coordination of the first two nodes is initiated with (0,0,0) and (1,0,0), respectively.
2. For i= 1 to n−1 do Step3.
3. Generated to X_i from 0 to 4 value according to the normal distribution.

2) Updating Individual: The new individual generation method applied two methods, Attract H and move pull methods, depending on the structure of protein sequences. Attract H method considers an intensification process. which is a very important for rapid convergence to the optimal solution. Besides, the move pull method also considers a diversification process, which generates alternative solutions to cover more regions in the search space. The Attract H and the move pull methods are used to generate new solutions in one individual mutation.

a) Attract H Method: Attach H method moves H node location beside other H node location If allowed, and this movement should make the energy of protein with lower values. This procedure is illustrated in Procedure 4.2.

Procedure 4.2: Attract H
1. Adjust all H nodes not adjacent to other H nodes or adjacent to at least one node and put them in the HH vector.
2. Search for an empty location, from the UDLRFB matrix with empty places adjacent to all H nodes.
3. For i=1 to the numeral of H nodes do Steps from Step 4 to Step 6.
4. Move the H node to be adjacent to any other H node if allowed by changing its location.
5.Make changes in the remaining nodes coordinate nodes to achieve the connectivity.
6. Finally, ensure that no overlapping prevention or expansion to accept the movement else refuse the solution.

Fig. 4 presents the effect of Attract H method on the p2 model; Fig. 4(a) shows the p2 model without using Attract H method, and Fig. 4(b) shows the p2 model after using Attract H method. There is a clear difference since energy has a lower value after applying Attract H method.

b) Move Pull Method: Move pull method is considered as an intensification process, focusing on the solution. Its function is to choose three nodes linked together, randomly and then move the three cells in all available directions as presented in Fig. 5 to find the best solution or to make an H node adjacent to another H node with no links between them. This and this improves the resulting solutions. Procedure 4.3 presents the method.

Procedure 4.3: Move Pull
1. Generate a random number rN from 2 to n − 3.
2. Choose three consecutive nodes S_{rN-1}, $S_{rN}$and S_{rN+1}.
3. Detect the recent conformation of these three nodes.
4. Change the conformation from the remaining conformation as presented in Fig. 5 randomly.
5. Change coordinate all remaining nodes until connectivity is achieved.

We generate new individuals, either through one cluster center or more, or through one individual or two. To know which to use,a random value between the (0,1) range is generated. There are two ways to generate the new individual, The first is from one cluster, with the following procedure 4.4 describes the first updating method:

Procedure 4.4: Updating individual
1. Generate random values in the (0,1) range.

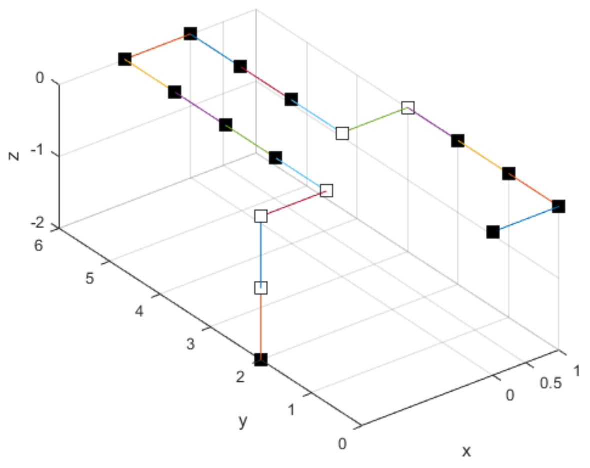


Fig. 3. Example in HP Lattice Model.

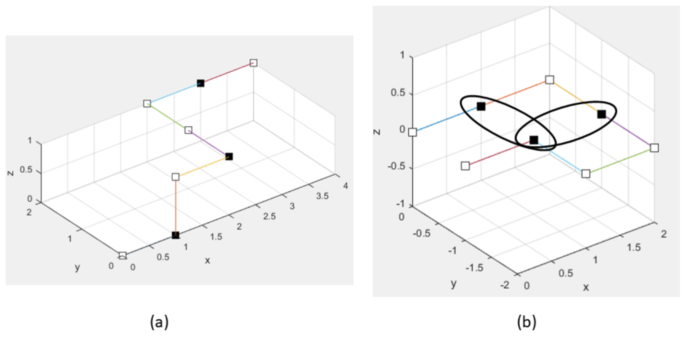


Fig. 4. Apply Attract H Algorithm on Simple Sample HP Model.

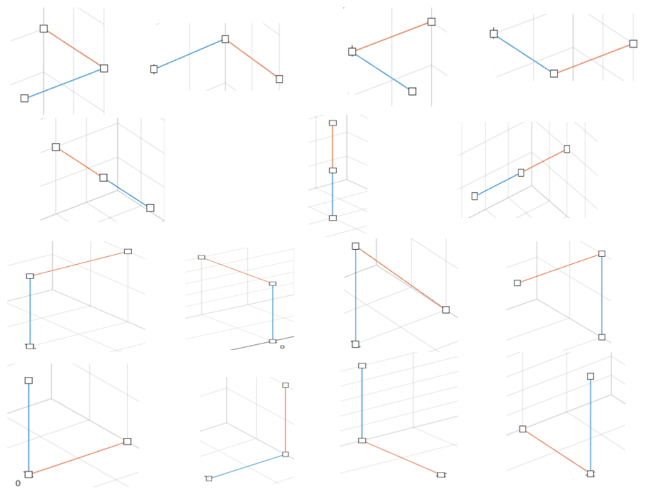


Fig. 5. Conformation on 3 Nodes.

2. If the value generated is less than the predetermined value, Then select one cluster center and update it by using Attract H method using procedure 4.2

3. Else, choose a random individual from the cluster group to update it by using Move Pull method using procedure 4.3.

The second part generates the new individuals from two cluster centers or two individuals of two different clusters. This method is considered a diversification process. The following procedure 4.5 describes the second new individual generation method:

Procedure 4.5: Updating individual
1. Generate random values in the (0,1) range.
2. If the value generated is less than the predetermined value, then select two random cluster centers and combine them using the crossover process.
3. Else, choose a random individuals from two clusters to combine the by crossover process.

V. Experiments and Discussion

BSO-HP algorithm is applied in different HP benchmark models [27, 28, 29] shown in Table XI.

A. Parameter Settings

All parameter values are summarized with their assigned values. These values have a common setting in the literature or are determined through our preliminary numerical experiments. Table X presents additional parameters applied to solve the HP model problem.

TABLE X. Additional Parameters for Solving HP Model Problem

Parameter Operators	parameters values	Description
Move pull parameter		
N_trial	10	Number of trial numbers
N_node	3	The number of the nodes that conformed
Penalty Parameters		
mu	1000	The penalty parameters.
eps	1e-5	The penalty parameters.
Termination parameter		
GenMax	5000 or when reach to the solution	The number of generations

B. Performance Analysis

The BSO-HP method is programmed in MATLAB. It is presented on 14 benchmark HP models, which are shown in Table XI. Table XI shows that our algorithm deals with different lengths of protein sequences.

TABLE XI. PROTEIN SEQUENCES

No.	length	protein sequence
P1	5	HPPHP
P2	8	PHPHPPHP
P3	13	HPPHPPHPHPPHP
P4	17	HHHHPPHHHHHHHPPPH
P5	20	HPHPPHHPHPPHPHHPPHPH
P6	21	PHPHPPHPHPPHPPHPHPPHP
P7	24	HHPPHPPHPPHPPHPPHPPHPPHH
P8	25	PPHPPHHPPPPHHPPPPHHPPPPHH
P9	27	HHHHPPPPPHPPPPPHHHPPPPPPPPH
P10	34	HPPHPPHPHPPHPPHPHPPHPHPPHPPHPHPPHP
P11	36	PPPHHPPHHPPPPPHHHHHHHPPHHPPPPHHPPHPP
P12	48	PPHPPHHPPHHPPPPPHHHHHHHHHHPPPPPPHHPPHHPPHPPHHHHH
P13	50	HHPHPHPHPHHHHPHPPPHPPPHPPPPHPPPHPPPHPHHHHPHPHPHPHH
P14	60	PPHHHPHHHHHHHHPPPHHHHHHHHHHPHPPPHHHHHHHHHHHHPPPPHHHHHHPHHPHP

Table XII presents the results of our proposed BSO-HP method. The best energy values founded in one run are recorded. These results emphasize that our method can find the best-known solution for all HP models except in p6 and p9 models; our method can also find the new optimal solution.

TABLE XII. BSO-HP RESULTS

HP	length	best sol.	HP-BSO
p1	5	-1	-1
p2	8	-2	-2
p3	13	-5	-5
p4	17	-9	-9
p5	20	-11	-11
p6	**21**	**-8**	**-9**
p7	24	-13	-13
p8	25	-9	-9
p9	**27**	**-9**	**-10**
p10	34	-19	-19
p11	36	-18	-18
p12	48	-29	-29
p13	50	-26	-26
p14	60	-49	-49

Sample results presented in Fig. 6 are obtained from different dimensions. For the problem, Fig. 6(b) and Fig. 6(c) obtain the best solution from all algorithms treated with this problem.

The strength of the BSO-HP method is in finding more than one construction of the same model with the optimal solution. Fig. 7 show how the BSO-HP method found multishapes of the best solution. Fig. 7(a) and Fig. 7(b) show multiconformation of sequence p3 model with length 13 and the energy is -5, Fig. 7(c) and Fig. 7(d) show multiconformation of sequence P4 model with length 17 and the energy is -9

C. Comparison Results

BSO-HP method was compared with other methods to exhibit the strength of the method. Table XIII presents the comparison between theBSO-HP method with MCMPSO-TS [28], HGA-PSO [29], and TPPSO [27] based on reaching the optimal solution. MCMPSO-TS method was tested on p1, p2, p3, p4, p5, p6, p8, p10, and p11 and focused on the small HP lengths. HGA-PSO method was tested on p5, p7, p8, p11, p12, p13, and p14. The TPPSO method that was tested on p9, p10. BSO-HP method covered all benchmark models and not only

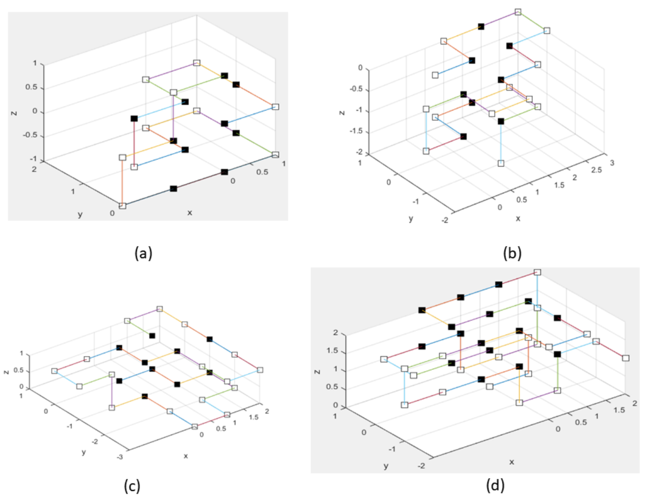


Fig. 6. Conformation of Sequence p5, p6, p10 and p12 Respectively.

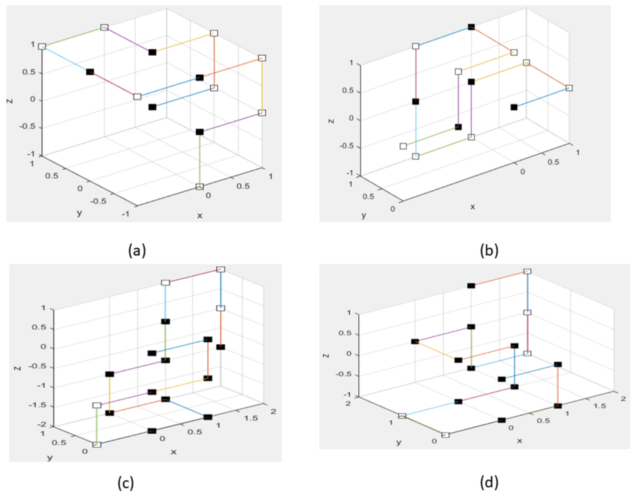


Fig. 7. (a) and (b) Multiconformation of Sequence P3 with Length 13 and the Energy is -5, (c) and (d) Multiconformation of Sequence P4 with Length 17 and the Energy is -9.

found the optimal solution in all models but also got the best solution compared with the rest methods.

VI. CONCLUSION

An adaptive discrete brainstorm algorithm is designed to deal with nonlinear integer programming problems and their applications. The BSO-IP algorithm used the prior knowledge of best solutions in the search space to generate new solutions. This convergence operator helps reach the optimal solution. Several sets of benchmark test problems of nonlinear integer programming problems were tested, and the results proved the promising performance of the BSO-IP. Additionally, the proposed method BSO-HP was applied to solve PSP problems as an NP integer programming problem. The BSO-HP algorithm employed the same procedures as the BSO-IP algorithm, except in some additional procedures to deal with the biological basis in the PSP problem. Numerical results

TABLE XIII. Comparison Between BSO-HP and Other Methods

HP	length	best	MCMPSO-TS	HGA-PSO	TPPSO	HP-BSO
p1	5	-1	-1	-	-	-1
p2	8	-2	-2	-	-	-2
p3	13	-5	-5	-	-	-5
p4	17	-9	-9	-	-	-9
p5	20	-11	-11	-11	-	-11
p6	21	-8	-8	-	-	**-9**
p7	24	-13	-	-13	-	-13
p8	25	-9	-9	-9	-	-9
p9	27	-9	-	-	-9	**-10**
p10	34	-19	-19	-	-	-19
p11	36	-18	-18	-18	-17	-18
p12	48	-29	-	-29	-	-29
p13	50	-26	-	-26	-	-26
p14	60	-49	-	-49	-	49

show that the BSO-HP method is a promising optimization method. Moreover, the BSO-HP method obtained new optimal solutions for two benchmark protein sequences. We will apply our proposed method to the other types of PSP problems as the 3D face-centred-cube HP model. Also, The proposed method obtained multishapes of the same protein sequence with the same lowest energy, a feature important to biologists. We would like to improve our proposed method to be able to help biologists in a laboratory.

References

[1] J Kennedy, R Eberhart, and Y Shi. *Swarm Intelligence*. Morgan Kaufmann Publisher, Burlington, 2001.

[2] K M Passion. Bacterial foraging optimization. *Int. J. Swarm Intell. Res*, 1(1):1–16, 2010.

[3] D Karaboga. An idea based on honey bee swarm for numerical optimization, 2005.

[4] M Dorigo and T Stützle. *Ant Colony Optimization*. The MIT Press, Cambridge, 2004.

[5] Y Shi. An optimization algorithm based on brainstorming process. *Int. J. Swarm Intell. Res*, 2:35–62, 2011.

[6] Y Shi, Y Tan, Y Shi, and Y Chai. Brain storm optimization algorithm. In Wang and G., editors, *ICSI 2011, Part I. LNCS*, volume 6728, pages 303–309. Springer, 2011.

[7] Alwaleed Aldhafeeri and Yahya Rahmat-Samii. Brain storm optimization for electromagnetic applications: continuous and discrete. *IEEE Transactions on Antennas and Propagation*, 67(4):2710–2722, 2019.

[8] Shoubao Su, Jiwen Wang, Wangkang Fan, and Xibing Yin. Good lattice swarm algorithm for constrained engineering design optimization. In *2007 International conference on wireless communications, networking and mobile computing*, pages 6421–6424. IEEE, 2007.

[9] Yingruo Xu, Yali Wu, Yulong Fu, Xinrui Wang, and Ating Lu. Discrete brain storm optimization algorithm based on prior knowledge for traveling salesman problems. In *2018 13th IEEE Conference on Industrial Electronics and Applications (ICIEA)*, pages 2740–2745. IEEE, 2018.

[10] W Ian, Laura Weston Davis, Jane S Murray, David C Richardson, and Richardson. Molprobity: structurevalidation and all-atom contact analysis for nucleic acids and their complexes. *Nucleic acids research*, 32(suppl2):615–619, 2004.

[11] Alice E Smith and David W Coit. Penalty functions.Handbook on Evolutionary Computation. *C*, 5:1–6, 1997.

[12] James Macqueen. *Some methods for classification and analysis of multivariate observations. InProceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1. Oakland, CA, USA, 1967.

[13] C Elena, Laskari, E Konstantinos, Michael N Parsopoulos, and Vrahatis. Particle swarm optimization for integer programming. *Computational Intelligence, Proceedings of the World on Congress on*, pages 1582–1587, 2002.

[14] Max Fogiel et al. The operations research problem solver, 1996.

[15] C Elena, Laskari, E Konstantinos, Michael N Parsopoulos, and Vrahatis. Particle swarm optimization for integer programming. *Computational Intelligence, Proceedings of the World on Congress on*, pages 1582–1587, 2002.

[16] Kusum Deep, Krishna Pratap Singh, M. L. Kansal, and C. Mohan. A real coded genetic algorithm for solving integer and mixed integer optimization problems. *Applied Mathematics and Computation*, 212(2):505–518, 2009.

[17] H R Paulo, Alexandre Cb Gabriel, and Delbem. Representations for evolutionary algorithms applied to protein-structure prediction problem using hp model. *InAdvances in Bioinformatics and Computational Biology*, pages 97–108, 2009.

[18] Chenhua Huang, Xiangbo Yang, and Zhihong He. Protein folding simulations of 2D HP model by the genetic algorithm based on optimal secondary structures. *Computational Biology and Chemistry*, 34(3):137–142, 2010.

[19] Shih-Chieh Su, Cheng-Jian Lin, and Chuan-Kang Ting. An effective hybrid of hill climbing and genetic algorithm for 2D triangular protein structure prediction. *Proteome Science*, 9(Suppl 1):S19–S19, 2011.

[20] Tamjidul Hoque, Madhu Chetty, and Abdul Sattar. Protein folding prediction in 3d fcc hp lattice modelusing genetic algorithm. InEvolutionary Computation. *IEEE Congress on*, pages 4138–4145, 2007.

[21] Andrea Bazzoli, G B Andrea, and Tettamanzi. A memetic algorithm for protein structure prediction in a3d-lattice hp model. *InApplications of Evolutionary Computing*, pages 1–10, 2004.

[22] Torsten Thalheim, Daniel Merkle, and Martin Middendorf. Protein folding in the hp-model solved with a hybridpopulation based aco algorithm. *IAENG International Journal of Computer Science*, 35(3):291–300, 2008.

[23] Xiaolong Zhang and Wen Cheng. An improved tabu search algorithm for 3d protein folding problem, 2008.

[24] Haitao ésar Rego, Fred Li, and Glover. A filter-and-fan approach to the 2d hp model of the protein foldingproblem. *Annals of Operations Research*, 188(1):389–414, 2011.

[25] Tamjidul Hoque, Madhu Chetty, and Abdul Sattar. Protein folding prediction in 3d fcc hp lattice modelusing genetic algorithm. InEvolutionary Computation. *IEEE Congress on*, pages 4138–4145, 2007.

[26] Lauro Luiz Fernando Nunes, Heitor Cesar Gal vao, Silv, Pablo Lopes, Regina Moscato, and Berretta. Aninteger programming model for protein structure prediction using the 3d-hp side chain model. *Discrete AppliedMathematics*, 198:206–214, 2016.

[27] Yuzhen Guo, Fengying Tao, Zikai Wu, and Yong Wang. Hybrid method to solve HP model on 3D lattice and to probe protein stability upon amino acid mutations. *BMC Systems Biology*, 11(S4):93–93, 2017.

[28] Changjun Zhou, Caixia Hou, Qiang Zhang, and Xiaopeng Wei. Enhanced hybrid search algorithm for protein structure prediction using the 3D-HP lattice model. *Journal of Molecular Modeling*, 19(9):3883–3891, 2013.

[29] Shih-Chieh Cheng-Jian Lin and Su. Protein 3d hp model folding simulation using a hybrid of genetic algorithmand particle swarm optimization. *International Journal of Fuzzy Systems*, 13(2), 2011.