International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Improved Euclidean Particle Swarm and Application

Hongbing Zhu[1*], **Chengdong Pu**[2], **Jinguang Gu**[2]

[1] *Hiroshima Kokusai Gakuin University, Japan*

[2] *Wuhan University of Science and Technology, China*

[*] *Corresponding author's Email: yekon7312@gmail.com*

Abstract: Particle swarm optimization (PSO) is a swarm intelligence algorithm, has been successfully applied to many engineering optimization problems and shown its high search speed in these applications. However, as the dimension and the number of local optima of optimization problems increase, PSO and most existing improved PSO algorithms such as, the standard particle swarm optimization (SPSO) and the Gaussian particle swarm optimization (GPSO), are easily trapped in local optima. In this paper we proposed a novel algorithm based on SPSO called Euclidean particle swarm optimization (EPSO) which has greatly improved the ability of escaping from local optima. To confirm the effectiveness of EPSO, we have employed five benchmark functions to examine it, and compared it with SPSO and GPSO. The experiments results showed that EPSO is significantly better than SPSO and GPSO, especially obvious in higher-dimension problems. As one of tis application, we applied EPSO to the structure prediction of toy model both on artificial and real protein sequences. Predicting the structure of protein through its sequence of amino acids is a complex and challenging problem in computational biology. Though toy model is one of the simplest and effective models, it is still extremely difficult to predict its structure as the increase of amino acids. The experimental results demonstrated that EPSO was efficient in protein structure prediction problem in toy model.

Keywords: Particle swarm optimization, Euclidean distance, Protein Structure Prediction, Toy Model

1. Introduction

Eberhart and Kennedy developed a particle swarm optimization (PSO) [1, 2] as a swarm intelligence algorithm [3, 4] simulating a swarms of birds schooling. It has been successfully employed to solve optimization problems effectively in various areas [5, 6, 7]. Shi and Eberhart put forward the conception of inertia weight to improve the performance of PSO, and this method called standard particle swarm optimization (SPSO) [8, 9]. Renato A. Krohling proposed Gaussian particle swarm optimization algorithm (GPSO) [10] based on the Gaussian probability distribution.

However, SPSO is easily trapped in local optima and premature convergence, especially in higher-di–mension problems. The main reason is that particles will fly to their own best position and the global particle's position rapidly according to their own and companion's flying experience [11]. So the particles will become more and more similar and cluster to the global best particle that falls into local optima [12].

In order to overcome the disadvantages of SPSO, we proposed a novel approach called Euclidean particle swarm optimization (EPSO). If the global best fitness has not been updated for a certain times, it may be trapped in local optima, so the velocities of particles will get an interference factor. The value of interference factor will be self-adaptive according to the Euclidean distance between the current particle and the global best particle. Experiments confirmed that EPSO kept the diversity of particles and has greatly improved the ability of escaping from local optima.

The structure of protein determines its function in molecular. Predicting the structure of protein through its sequence of amino acids is a complex and challenging problem in computational biology. The native structure of a protein is associated with the structure of the global minimum of the free energy consisting of the intramolecular interaction among protein atoms and between the proteins and surrounding solvent molecules [14]. Based on this minimum free-energy theory, many simplified protein models have been proposed to predict the structure of protein. Toy model is one of the simplest and most effective protein models proposed by Stillinger 1993 [15], however, it is still extremely difficult to predict the structure of protein with it. Many methods, such as high temperature Monte Carlo method (HTML) [16], pruned enriched Rosenbluth method (PERM) [17] and Particle Swarm Optimization with a Constriction Factor (CPSO) [18], have applied to search ground state structure of protein based on toy model.

In this paper, we will apply EPSO to protein structure prediction of toy mode both on artificial and real protein sequence and present a lot of experimental results to verify its effectiveness.

2. Standard Particle Swarm Optimization

SPSO is also a population-based algorithm which initialized with a population of candidate solutions. Each candidate solution in SPSO is called a particle. Each particle has a velocity vector $\mathbf{v}_i = (v_{i1}, v_{i2} ... v_{in})$ and a position vector $\mathbf{x}_i = (x_{i1}, x_{i2} ... x_{in})$. Each particle knows its best fitness f_{pbest} and position $\mathbf{p}_i = (p_{i1}, p_{i2} ... p_{in})$ so far. Moreover, each particle knows the global best fitness f_{gbest} in the group among f_{pbest} and position $\mathbf{p}_g = (p_{g1}, p_{g2} ... p_{gn})$ so far. Here $i = 1, 2, ...m$ is the particle's index, m is the total number of particles, and n is the dimension of optimization problem. Velocity and position of each particle can be modified by the following equations:

$$\mathbf{v}_i = w\mathbf{v}_i + c_1 rand(\mathbf{p}_i - \mathbf{x}_i) + c_2 Rand(\mathbf{p}_g - \mathbf{x}_i) \quad (1)$$

$$\mathbf{x}_{i+1} = \mathbf{x}_i + \mathbf{v}_i \quad (2)$$

where w is the inertia weight, usually decreasing linearly from 0.9 to 0.4 [9]; c_1 and c_2 are positive constants, called acceleration coefficients, usually setting $c_1 = c_2 = 2.0$; *rand* and *Rand* are random numbers in the range [0,1] generated according to a uniform probability distribution. Particles' velocities along each dimension are limited to v_{max} which is specified by user.

3. Euclidean Particle Swarm Optimization

In view of the defects of SPSO, we proposed an improved algorithm EPSO. The core concept of EPSO is that if the global best fitness has not been updated for K times, velocities of particles will get an interference factor to make most of particles fly out of the local optima but the best one is kept continuing to do local search. Here K is a constant which will be determined by experiments described in section IV(C).

The value of interference factor is related to the current particle's position $\mathbf{x}_i$ and the global best particle's position $\mathbf{p}_g$ and produced by sigmoid function according to the Euclidean distance between $\mathbf{x}_i$ and $\mathbf{p}_g$. So the interference factor is called Euclidean interference factor ε that is obtained by modified sigmoid function as below:

$$\varepsilon_i = (\frac{1}{1 + \exp(-a/d_i)} - 0.5) * 2v_{max} \quad (3)$$

where d_i is the Euclidean distance from current particle i to the global best particle; slope parameter a usually set to 0.5.

We add ε to velocities of particles if the global best fitness has not been updated for K times, and update the velocities of particles as following:

$$\mathbf{v}_i = w\mathbf{v}_i + c_1 rand(\mathbf{p}_i - \mathbf{x}_i) + c_2 Rand(\mathbf{p}_g - \mathbf{x}_i) + \varepsilon_i \quad (4)$$

In the initial stage of optimizing, the value of d is large, ε will be small, particles will continue to search along original directions; in the late stage of optimizing, the value of d is very small due to the high degree of similarity of particles, and ε will be large, so particles will get a large interference and escape from local optima to find a better fitness.

The algorithm of EPSO is described as follows:

Step 1 Initial the value of K and maximum iteration number. Initial the position $\mathbf{x}_i$ and the velocity $\mathbf{v}_i$ of each particle within the allowable range.

Step 2 Calculate the fitness of each particle.

Step 3 Determine the previous best position $\mathbf{p}_i$ and fitness f_{pbest} of every particle.

Step 4 Determine the global best position $\mathbf{p}_g$ and fitness f_{gbest} so far. If f_{gbest} has not been updated, counter $c = c + 1$.

Step 5 If $c > K$, update $\mathbf{v}_i$ and $\mathbf{x}_i$ according to (6), (7) and (4), and then set the counter $c = 0$. Else, update $\mathbf{v}_i$ and $\mathbf{x}_i$ according to (5), (6).

Step 6 If the maximum iteration number is met, stop algorithm, else go to **Step 2**.

4. Experiments and Discussion

4.1 Benchmark functions

In order to confirm the effectiveness of EPSO, five benchmark functions, which were popularly used in the literatures [1, 8, 10], will be also used in our experiments. The five functions are shown as below.

F_1: Sphere function

$$f(x) = \sum_{i=1}^{n} x_i^2,\; x_i \in [-100, 100]$$

F_2: Rosenbrock function

$$f(x)=\sum_{i=1}^{n}(100(x_{i+1}-x_i^2)^2+(x_i-1)^2),\; x_i \in [-30,30]$$

F_3: Rastrigin function

$$f(x)=\sum_{i=1}^{n}(x_i^2-10\cos(2\pi x_i)+10),\; x_i \in [-5.12,5.12]$$

F_4: Griewank function

$$f(x)=\frac{1}{4000}\sum_{i=1}^{n}x_i^2-\prod_{i=1}^{n}\cos(\frac{x_i}{\sqrt{i}})+1,\; x_i \in [-600,600]$$

F_5: Schaffer's f6 [13] function

$$f(x)=0.5-\frac{(\sin\sqrt{x_1^2+x_2^2})^2-0.5}{1.0+0.001(x_1^2+x_2^2)^2},\; x_i \in [-100,100]$$

4.2 Experimental setting

All the experiments were performed on a computer with Intel 2 core 2.5G processor, 2G memory, Linux-64 system. All the algorithms were written in *C* and compiled by gcc-4.3.2 compiler. In all cases, the swarm size was 20, inertia weight w was 0.9 at the beginning of the run, and made to decrease linearly to 0.4 at the end. The parameters c_1 and c_2 were 2.0. The maximum velocity v_{max} was set at half value of the upper bound.

4.3 Determination of parameter *K*

The value of parameter K is very important for the performance of EPSO algorithm. We took three benchmark functions to determine the value of K. The dimensions of those functions were set to be 20 and 30. Let K increase from 0 to 1500 (+5), and computed the success convergence rate of 100 running times in the maximum iteration number. The convergence error and the maximum iteration number for each function were different. The results are shown in Figure 1.

Figure 1 shows that EPSO has higher convergence rate for those benchmark functions when K got the value on interval [20,100]. So the value of K in EPSO is set as 60 which is the sharp middle number of interval [20,100].

4.4 Comparison of SPSO, GPSO and EPSO

To confirm the effectiveness and performance of EPSO, we compared it with SPSO and GPSO based on the benchmark functions introduced in section IV(*A*). For the purpose of comparison, the experimental parameters were the same as the section IV(*B*).

The functions F_1 to F_4 were used to find the minimum 0. The dimensions of each function were set to be 10 to 50 (+10), the maximum numbers of iterations were different according to the difficulty to find the minimum respectively. Each experiment was repeated for 1000 times to compare the performance of EPSO with SPSO and GPSO through the mean best fitness and the smallest best fitness that they obtained for all the runs. The results were listed in Table 1.

In Table 1, Dim. is the dimensions of functions, Gen. is the maximum numbers of iteration, Mean and Best are the mean value and smallest value of best fitness after the algorithm run Gen. times, and B.rate is the better rate of mean best fitness compared EPSO with GPSO or SPSO.

As seen in Table 1, EPSO is significantly better than SPSO and GPSO both in the benchmark functions. EPSO found the lower Mean and Best value of functions, and outperformed the other two algorithms. Although, EPSO is worse than GPSO in Rastrigin function, the B.rate is decreasing as the increment of function's dimensions and it is still better than SPSO obviously. Furthermore, Table 1 also shows that the bigger dimensions of functions are, the better performance of EPSO will have.

In order to further confirm the performance of EPSO, we also took F_5 for experiment. F_5 was a benchmark function in Genetic Algorithm. It would always fall into the same local optima in finding the maximum, no matter how many the maximum iteration number was. This experiment was to compare the performance of EPSO with SPSO and GPSO through the times of finding the maximum 1.0 after repeated for 1000 times. The maximum numbers of iteration set as

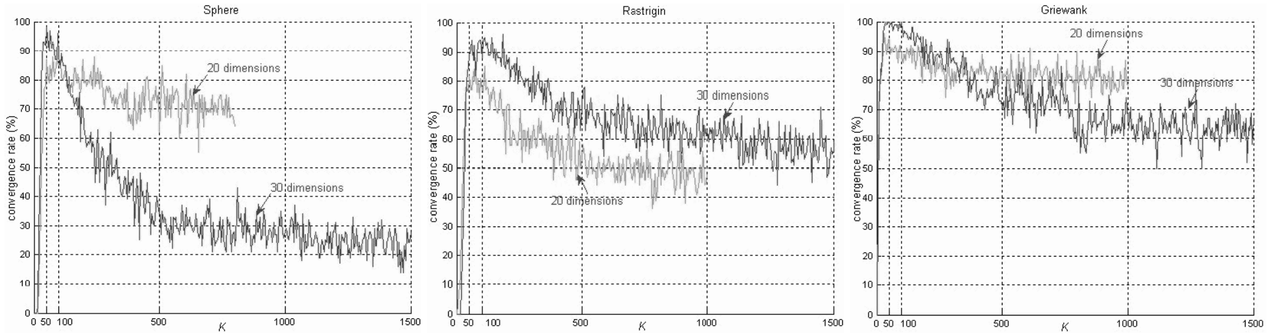


Figure 1 Experiments results for determining K

Table 1 The experimental results for benchmark functions and comparison

			EPSO		GPSO			SPSO		
	Dim.	Gen.	Mean	Best	Mean	Best	B.rate	Mean	Best	B.rate
F_1	10	500	0.0000	0.0000	0.0000	0.0000	0%	0.0000	0.0000	0%
	20	1000	0.0000	0.0000	0.0005	0.0000	100%	0.0359	0.0000	100%
	30	1500	0.0000	0.0000	0.3477	0.0000	100%	0.9024	0.0000	100%
	40	2000	0.0000	0.0000	2.8394	0.0000	100%	4.2047	0.0000	100%
	50	2500	0.0000	0.0000	17.1798	0.0000	100%	22.7763	0.0001	100%
F_2	10	1000	2.2827	0.0000	3.4794	0.0000	34%	2.4215	0.0000	6%
	20	3000	12.5966	0.0000	15.4443	0.0001	18%	21.4682	0.0000	41%
	30	5000	33.1653	0.0000	147.7717	0.0019	78%	218.4175	0.0000	85%
	40	7000	51.8153	0.0001	302.5478	0.1330	83%	594.7099	0.0004	91%
	50	9000	69.8725	0.0202	3852.2849	10.2961	98%	3091.4274	0.6425	98%
F_3	10	1000	3.7578	0.0000	5.3399	0.0000	30%	7.4054	0.0000	49%
	20	3000	9.1287	1.9899	8.2607	0.0000	-11%	17.0971	2.9848	47%
	30	5000	18.9856	4.9748	11.9051	0.9949	-59%	29.2762	6.3722	35%
	40	7000	30.9724	8.9546	22.3094	2.9854	-39%	45.7170	13.0730	32%
	50	9000	44.9705	16.9143	41.8352	10.0307	-7%	69.1484	21.9986	35%
F_4	10	500	0.1391	0.0000	0.1419	0.0073	2%	0.1469	0.0000	5%
	20	1000	0.0412	0.0000	0.0441	0.0000	7%	0.0807	0.0000	49%
	30	1500	0.0180	0.0000	0.0438	0.0000	59%	0.2192	0.0000	92%
	40	2000	0.0123	0.0000	0.1602	0.0000	92%	0.4104	0.0000	97%
	50	2500	0.0094	0.0000	0.5274	0.0000	98%	0.7113	0.0002	99%

1500. The results were listed in Table 2.

Table 2 The experimental results for F_5

Algorithms	Mean	Best	Obtained times
SPSO	0.9928	1.0000	237
GPSO	0.9928	1.0000	237
EPSO	0.9980	1.0000	807

As shown in Table 2, each algorithm could find maximum 1.0, but EPSO obtained for 807 times, far more than 237 times which SPSO and GPSO obtained. It shows that EPSO easily escape from local optima, and find the optimum of function.

To sum up the above arguments, EPSO has better convergence efficiency and precision than SPSO and GPSO. It has greatly improved the ability of escaping from local optima to find the better fitness, especially obvious in higher-dimension problems.

5. Toy model

In the toy model, the 20 amino acid residues are classified into hydrophobic residues and hydrophilic residues represented by the letters A and B respectively. There is only one fixed length bond between two consecutive residues. And the angle between the two bonds can change freely. The configuration of any n-mer is specified by the $n-2$ angles of bend $\theta_2, ..., \theta_{n-1} \in [-\pi, \pi)$, residues along the backbone can be encoded by a set of binary variables $\xi_1, ..., \xi_n$. If $\xi_1 = 1$, the ith residue is A, if $\xi_1 = -1$, it is B. The intramolecular protein energy function Φ is expressed as follows for any n-mer :

$$\Phi = \sum_{i=2}^{n-1} V_1(\theta_i) + \sum_{i=1}^{n-2} \sum_{j=i+2}^{n} V_2(r_{ij}, \xi_i, \xi_j) \quad (5)$$

where $V_1(\theta_i)$ is backbone bend potentials, expressed as formula (6) and $V_2(r_{ij}, \xi_i, \xi_j)$ is nonboned interactions, expressed as formula (7).

$$V_1(\theta_i) = 1/4(1 - \cos\theta_i) \quad (6)$$

$$V_2(r_{ij}, \xi_i, \xi_j) = 4(r_{ij}^{-12} - C(\xi_i, \xi_j) r_{ij}^{-6}) \quad (7)$$

where r_{ij} denotes the distance between residue i and j of the chain. The coefficient $C(\xi_i, \xi_j) = (1 + \xi_i + \xi_j + 5\xi_i\xi_j)$. For an AA pair, $C(\xi_i, \xi_j) = 1$, regarded as strongly attracting, for a AB pair, $C(\xi_i, \xi_j) = 0.5$, regarded as weakly attracting and for a BB pair, $C(\xi_i, \xi_j) = -0.5$, regarded as weakly repelling. The protein structure prediction problem can be described as giving a sequence of amino aid residues to find a group solution to make the energy Φ minimal.

6. Experiments and Discussion

To confirm the effectiveness and performance of EPSO applied in protein structure prediction of toy model, we took experiments both on artificial and real protein sequences, and compared the results with reported in other papers.

All the experiments were implemented by C in Linux system. In all cases, the swarm size was 20, K was 60, inertia weight w was 0.9 at the beginning of the run, and made to decrease linearly to be 0.4 at the maximum number of iterations. The parameters c_1 and c_2 were 2.0. The max velocity v_{max} was set at half value of the upper bound π.

6.1 Experiments on Artificial Sequences

For short protein sequences, EPSO got the ground state energy presented by Stillinger easily. Fibonacci sequences is the artificial protein sequence studied in Refs [16], they are defined recursively by:

$$S_0 = A, S_1 = B, S_{i+1} = S_{i-1} * S_i \quad (8)$$

where "$*$" is a concatenation operator. The first few sequences are $S_2 = AB$, $S_3 = BAB$, $S_4 = ABBAB$, etc. Hydrophobic residue A occurs isolated along the chain, while hydrophilic residue B occurs either isolated or in pairs and the molecules have a hierarchical string structure. We considered the Fibonacci sequences with length 13, 21, 34 and 55 for experiments, and the four artificial protein sequences are shown in Table 3.

Table 3 Fibonacci sequences

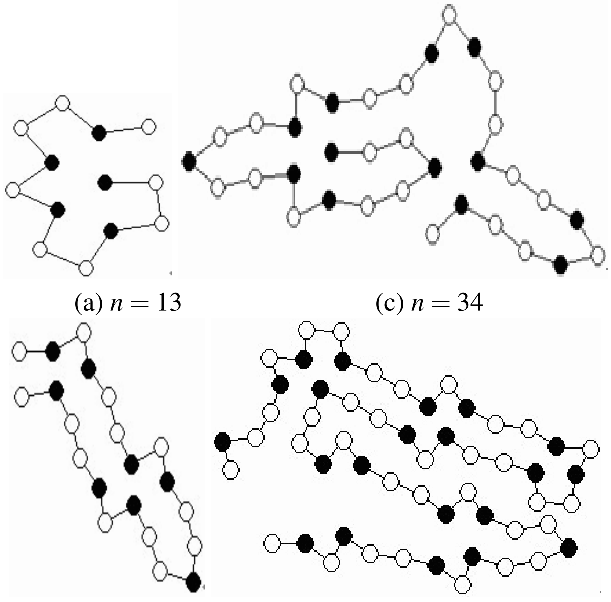
A.S.	Sequence
S_{13}	*ABBABBABABBAB*
S_{21}	*BABABBABABBABBABABBAB*
S_{34}	*ABBABBABABBABBABABBABABBABBABABBAB*
S_{55}	*BABABBABABBABBABABBABABBABBABABBAB* *BABABBABABBABBABABBAB*

where A.S. denotes the artificial protein sequence. The results of experiments are listed in Table 4. E_{HTML} is the minimum energy obtained by high temperature Monte Carlo method (HTML) [16], E_{PERM} is the lowest energy by pruned enriched Rosenbluth method (PERM) [17], E_{EPSO} is the lowest energy obtained by EPSO.

Table 4 Results on artificial sequence

A.S.	E_{HTML}	E_{PERM}	E_{EPSO}
S_{13}	-3.2235	-3.2167	-3.2941
S_{21}	-5.2881	-5.7501	-6.1980
S_{34}	-8.9749	-9.2195	-9.8341
S_{55}	-14.4089	-14.9050	-16.4474

From Table 4, it is clear that EPSO significantly better than HTML and PERM methods as it got much lower energy of the four protein sequences. Figure 2 shows the lowest energy conformations of those proteins obtained by EPSO, in which the black dots represent hydrophobic residues A and the white circles represent hydrophilic residues B.



(a) $n = 13$ (c) $n = 34$

(b) $n = 21$ (d) $n = 55$

Figure 2 The conformations of S_{13}, S_{21}, S_{34} and S_{55} obtained by EPSO

As shown in Figure 2, the hydrophobic residues form the clusters of particles, and are always flanked by hydrophilic residues along the chain. They simulate real protein structure in a certain degree. Especially for the conformation of S_{13}, it has the single hydrophobic core, which is analogous to the real protein structure perfectly in two-dimensions.

6.2 Experiments on Real Sequences

To further confirm the performance of EPSO in protein structure prediction, we took several real proteins for experiments. All information of these proteins were downloaded from PDB (http://www.r-csb.org/pdb/), and the real protein sequences are showed in Table 5.

Table 5 Real protein sequences

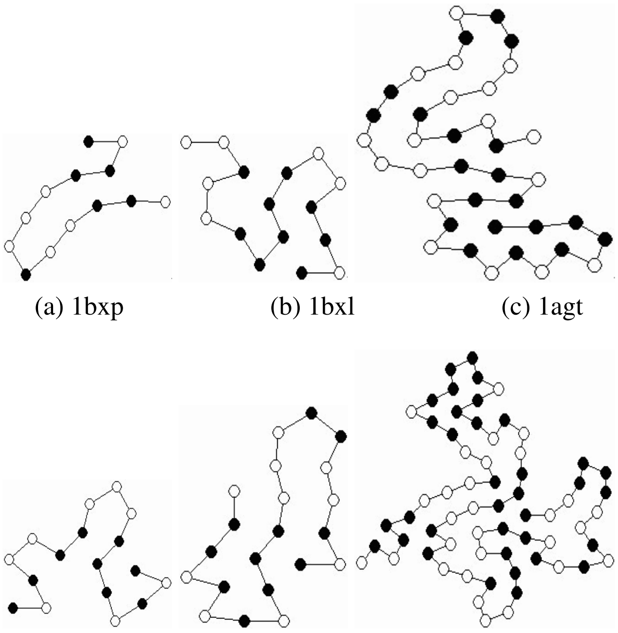
R.S.	Sequence
1bxp	*MRYYESSLKSYPD*
1bxl	*GQVGRQLAIIGDDINR*
1edp	*CSCSSLMDKECVYFCHL*
1edn	*CSCSSLMDKECVYFCHLDIIW*
1agt	*GVPINVSCTGSPQCIKPCKDQGMRFGKCMNRK* *CHCTPK*
1aho	*VKDGYIVDDVNCTYFCGRNAYCNEECTKLKGE* *SGYCQWASPYGNACYCYKLPDHVRTKGPGRCH*

where R.S. denotes the real protein sequence. We searched the minimal energy of these real proteins by EPSO and made comparison with other methods. The results are listed in Table 6. E_{GAA} and E_{LAGAA} are the optimum energy obtained by GAA and LAGAA respectively [25, 26], E_{SA} is the minimum energy obtained by Simulated Annealing Algorithm (SA) [24], E_{CPSO} is the lowest energy obtained by Particle Swarm Optimization with a Constriction Factor (CPSO) [18], E_{EPSO} is obtained by EPSO algorithm.

From Table 6, the minimal energy of those real protein obtained by EPSO is obviously lower than that by other algorithms. Figure 2 shows the lowest energy conformations of these proteins by EPSO.

Table 6 Results on real proteins

R.S.	E_{GAA}	E_{LAGAA}	E_{EPSO}
1bxp	-2.24484	-2.24484	-4.392713
1bxl	-8.74685	-8.81260	-8.812603
1edp	-5.60713	6-64530	-10.06692
1edn	-7.09609	-7.81925	-11.13420
R.S.	E_{SA}	E_{CPSO}	E_{EPSO}
1agt	-17.362815	-19.616866	-21.424246
1aho	-14.961273	-15.191101	-31.221805


(a) 1bxp (b) 1bxl (c) 1agt

(d) 1edp (e) 1edn (f) 1aho

Figure 3 The conformations of real proteins obtained by EPSO

As shown in Figure 3, the hydrophobic residues form the clusters of particles, and are flanked by hydrophilic residues along the chain. They are analogous to the real protein structure in two dimensions, that is to say EPSO is a great method to predict the structure of real protein in toy model.

7. Conclusion

In this paper, we proposed an original improved algorithm EPSO based on giving velocities of particles a Euclidean interference factor when the global best fitness has not improved for 60 generations, but keep the best one continuing to do local search. In order to confirm the effectiveness and performance of EPSO, we compared it with SPSO and GPSO algorithms based on benchmark functions. Experimental results show that EPSO has better convergence efficiency and precision, and has greatly improved the ability of escaping from local optima.

And we applied EPSO to the problem of protein structure prediction in toy model. We took some experiments both on artificial and real protein sequences to search their minimal energy and compared it with other methods. The experimental results showed that EPSO found the lower energy of protein and better than other methods. The conformations of proteins obtained by our method also confirm that EPSO is an effective method in protein structure prediction.

Acknowledgment

This work was partially supported by a grant from the NSF (Natural Science Fundation) of educational agency of Hubei Prov. under grant number D20091106 and the NSF of China under grant number 60803160.

References

[1] Kennedy J,Eberhart R., "Particle Swarm Optimization", *Proceeding of IEEE International Conference on Neural Networks (ICNN'95) Perth, IEEE Press, Australia*, pp.1942-1948, 1995.

[2] Eberhart, R. C., Kennedy, J. "A New Optimizer Using Particle Swarm Theory", *Proceedings of the Sixth International Symposium on Micro Machine and Human Science, Nagoya, Japan*, pp.39-43, 1995.

[3] Bonabeau E, Dorigo M, Theraulaz G, Swarm Intelligence: From Natural to Artificial Systems, *Oxford University Press*, England, 1999.

[4] Kennedy J, Eberhart R, Swarm Intelligence, *Morgan Kaufmann,San Mateo*, 2001.

[5] Kwang Y. Lee, Mohamed, Modern Heuristic Optimization Techniques, *IEEE, Piscataway*, 2008.

[6] Eberhart R, Simpson P, Computational Intelligence PC Tools, *Boston: Academic Press Professional*, 1996.

[7] Naka S, Genji T, Yura T, Fukuyama Y, "A Hybrid Particle Swarm Optimization for Distribution State Estimation", *IEEE Trans Power Systems, IEEE Press*, pp.60-68, 2003.

[8] Shi Y,Eberhart R, "A Modified Particle Swarm Optimizer", *Proceedings of IEEE International Conference on Evolutionary Computation, IEEE Press*, pp.69-73, 1998.

[9] Shi Y, Eberhart R, "Parameter Selection in Particle Swarm Optimization", *Proceedings of Annual Conference on Evolutionary Programming, MIT Press, San Diego*, 1998.

[10] R. A. Krohling, "Gaussian Swarm: a Novel Particle Swarm Optimization Algorithm", *Proceedings of the IEEE Conference on Cybernetics and Intelligent Systems, Singapore*, pp.372-376, 2004.

[11] Andrew, Stacey, Mirjana, JancicIan, Grundy, "Particle Swarm Optimization with Mutation", *Evolutionary Computation*, pp.1425-1430, 2003.

[12] P.Angeline, "Evolutionary Optimization Versus Particle Swarm Optimization: Philosophy and Performance Difference", *Proceedings of the Evolutionary Programming Conference, San Diego, USA*, pp.169-173, 1998.

[13] Davis, L., Ed, "Handbook of Genetic Algorithms", *New York, NY:Van Nostrand Reinhold*, 1991.

[14] C. B. Anfinsen, "Principles That Govern the Folding of Protein Chains", *Science*, pp.223-227, 1973.

[15] F. H. Stillinger, T. Head-Gordon and C. L. Hirshfel, "Toy Model for Protein Folding", *Phys. Rev.*, 1469-1477, 1993.

[16] F. H. Stillinger, "Collective Aspects of Protein Folding Illustrated by a Toy Model", *Physical Review E*, pp.2872-2877, 1995.

[17] H. P. Hsu, V. Mehra, "Structure Optimization in an Off-lattice Protein Model", *Phys. Rev.*, 2003.

[18] J. Liu, L. H. Wang, L. L. He, and F. Shi, "Analysis of Toy Model for Protein Folding Based on Particle Swarm Optimization Algorithm", *ICNC*, PP.636-645, 2005.

[19] Kennedy J, Eberhart R, "Particle Swarm Optimization", *Proceedings of IEEE International Conference on Neural Networks (ICNN'95)Perth, IEEE Press, Australia*, pp.1942-1948, 1995.

[20] Eberhart, R. C., Kennedy, J., "A New Optimizer Using Particle Swarm Theory", *Proceedings of the Sixth International Symposium on Micro Machine and Human Science, Nagoya, Japan*, pp.39-43, 1995.

[21] Shi Y, Eberhart R, "A Modified Particle Swarm Optimizer", *Proceedings of IEEE International Conference on Evolutionary Computation (ICEC'98), IEEE Press, Anchorage*, pp.69-73, 1998.

[22] Shi Y, Eberhart R, "Parameter Selection in Particle Swarm Optimization", *Proceedings of the 1998 Annual Conference on Evolutionary Programming, MIT Press, San Diego*, 1998.

[23] Hongbing Zhu, Chengdong Pu, "Euclidean Particle Swarm Optimization", *2nd International Conference on Intelligent Networks and Intelligent Systems*, 2009.

[24] L. Wang, "Perspective Roles of Short-and Longrange Interactions in Protein Folding", *Wuhan University Journal of Natural Sciences*, pp.182-187, 2004.

[25] Z. Xiaolong, L. Xiaoli., "Effective Protein Folding Prediction Based on Genetic-annealing Algorithm in Toy Model", *Workshop on Intelligent Computing and Bioinformatics of CAS*, pp.21-26, 2006.

[26] Z. Xiaolong, L. Xiaoli, "Protein Folding Prediction Using an Improved Genetic-Annealing Algorithm", *The 19th Australian Joint Conference on Artificial Intelligence*, pp.1196-1200, 2006.