http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 12 (15): 3058-3066, 2013
ISSN 1812-5638 / DOI: 10.3923/itj.2013.3058.3066

A Self-Adaptive Hybrid Intelligent Optimization Algorithm

Rong Li and Hong-Bin Wang
Department of Computer, Xinzhou Teachers' University, Xinzhou, China

Abstract: To better solve high-dimensional function optimization problems, for the defects of some optimization algorithms, such as premature convergence and low accuracy in Particle Swarm Optimization (PSO) and slow convergence in Bacterial Foraging Algorithm (BFA), this study presents a self-adaptive hybrid intelligent optimization algorithm based on BFA and PSO (ABSO for short). The ABSO algorithm first realizes dynamic no-linear self-adaptive improvement for learning factors and inertial weight of PSO and chemotaxis step length of BFA, respectively. After chemotaxis operation of BFA being finished, the optimization updating mechanism of PSO is introduced to continue to update bacterial location to help BFA escape from local optima which combines organically the optimization update mechanism of PSO and the chemotaxis update mechanism of BFA and well balances the global search and local development capabilities. Simulation results on four benchmark functions show that the ABSO algorithm is superior to BFA, PSO, self-adaptive PSO and other two kinds of BFA hybrid algorithm in convergence speed, accuracy and robustness. This proves the validity of the ABSO algorithm in high-dimensional function optimization problems.

Key words: Bacterial foraging algorithm, particle swarm algorithm, self-adaptive adjustment, variable step-length

INTRODUCTION

In recent years, the heuristic optimization algorithm based on biological intelligence has become one of the mainstream algorithm for those non-linear, non-differential, multi-peak and complex problems. Bacterial foraging optimization algorithm (BFA) is a kind of new bio-inspired optimization algorithm which was proposed by Passino, a professor at Ohio State University (Passino, 2002). Due to its advantages such as swarm intelligence parallel search and easy to maintain population diversity, BFA has become another hot spot in the research field of bionic algorithm. But the study found that the randomness characteristics of bacterial chemotaxis may result in slower chemotaxis and the accuracy of optimal solution is not high (Dasgupta *et al*., 2009). In order to improve the performance of BFA, researchers have proposed a variety of different methods, including adjusting algorithm parameters and mixing other intelligent algorithms.

In terms of parameter adjustment (Majhi *et al*., 2009) applied the BFO model of automatic chemotactic step to the neural network training; (Datta *et al*., 2008) designed a BFO algorithm of a self-adaptive chemotaxis step length according to a self-adaptive Delta modulation principle; (Chen *et al*., 2008) proposed a self-adaptive collaborative bacterial foraging algorithm based on biological self-adaptive search strategy. What these algorithms have in common is that from the perspective of chemotactic step-length, according to the amount of harvesting energy within the foraging life cycle, bacteria can self-adaptively adjust chemotactic step-length and without increasing the complexity of algorithm, the bacterial optimization efficiency can be improved.

In terms of the design of algorithm fusion, usually with the aid of differences between the internal operation mechanism of distinct intelligent bionic algorithms, other algorithms, especially Particle Swarm Optimization (PSO), are blended into BFA. Biswaw *et al*. (2007) and Yang *et al*. (2012) used the method of moving particles in PSO to completely replace bacterial chemotaxis operation and proposed a hybrid optimization algorithm to apply to multimodal function optimization (BSOA for short). Korani (2008) and Zhao *et al*. (2010) used particles moving speed in PSO to replace the bacterial chemotaxis tumble direction (BF-PSO for short); (Bakwad *et al*., 2009; Tang *et al*., 2007; Chu *et al*., 2008) introduced the basic algorithm of PSO into BFO algorithm and put forward the proposed bacterial swarming and fast bacterial swarming algorithm, respectively. Biswas *et al*. (2007) incorporated crossover and mutation operation of differential evolution algorithm to BFO algorithm and presented a hybrid global optimization algorithm. Biswas *et al*. (2007) updated the bacterial position with the iterative formula of PSO but chemotaxis operator did not play a role; Dasgupta introduced crossover and mutation operators in bacterial evolution and increased the diversity of population but it is also easy to cause the lack of excellent individual.

Corresponding Author: Rong Li, Department of Computer, Xinzhou Teachers' University, Xinzhou, China

In view of the above situation, in order to further improve the optimization performance of algorithm, this srudy takes into account both parameter adjustment and fusion of algorithm and proposes a self-adaptive hybrid optimization algorithm based on bacterial foraging algorithm and particle swam optimization algorithm (ABSO). The new algorithm introduces the dynamic no-linear self-adaptive improvement for learning factors and inertial weight of PSO and chemotaxis step-length of BFA, respectively. After the variable step-length chemotaxis operation being finished, the self-adaptive algorithm still uses the optimization update mechanism of PSO to continue to change bacterial position, so as to help BFA jump out of local optimum, thus balancing the local search capability and global development capability. Function tests show that compared with BFA, PSO, self-adaptive PSO, BSOA (Biswaw *et al*., 2007; Yang *et al*., 2012), BF-PSO (Korani, 2008; Zhao *et al*., 2010), the new algorithm improves the convergence speed and accuracy and has better optimization performance.

RELATED BASIS

Bacterial foraging optimization algorithm: Bacterial foraging algorithm is a kind of bionics random search algorithm based on bacterial foraging behavior, including three procedures: chemotaxis, reproduction and elimination-dispersal.

Chemotaxis is the core operation of BFA algorithm and it corresponds to the direction selection strategy taken in the process of bacterial foraging which has extremely important influence on convergence of the algorithm. Usually in the chemotaxis process, bacterial motor pattern involves tumble and swim. Tumble is the operation that bacterial move unit step length in arbitrary directions, its updated equation is as follows:

$$\theta^i(j+1,k,l)=\theta^i(j,k,l)+c(i)\Delta(i)/\sqrt{\Delta^T(i)\Delta(i)} \quad (1)$$

where, $\theta^i(j,k,l)$ represents the position in which bacterium i is in the ith generation chemotaxis, the kth reproduction and the lth elimination-dispersal operation, c(i) refers to step-length and $\Delta(i)$ indicates tumble direction. Whenever bacteria complete tumbling, the algorithm will judge whether the fitness value has been improved, if improved, bacteria will continue to move several steps along the same direction. Follow the circle until the fitness is no longer improved, or until the swim step reaches a designated threshold. This process is defined as swim. Thus the essence of bacterial foraging chemotaxis is that bacteria constantly search in the feasible solution neighborhood and determines to continue to swim along the direction, or re-adjust the direction. Chemotaxis operator ensures bacterial individual can find the optimal value within the neighborhood.

After a Chemotaxis cycle, bacteria enter the reproductive process of survival of the fittest. BFA adopts health degree as the criterion of evaluating the advantages and disadvantages of various bacteria, the computation formula is defined as follows:

$$J^i_{health}=\sum_{j=1}^{N_c+1}J^i(j,k,l) \quad (2)$$

In the above Eq. 2, J^i_{health} represents health degree of the ith bacterium, the greater J^i_{health} is, the worse bacterial health is. N_c refers to maximum times of chemotaxis procedure. The algorithm sorts all the S bacteria in ascending order according to their health degree and reproduces the better first half S_r (S_r=S/2) bacteria to replace the worse latter half ones, so as to speed up the search. Reproduction process has certain effect on the improvement of optimization accuracy.

The bacterial colony begins to migrate after the production of N_{re} times. Elimination-dispersal operation randomly initializes bacteria according to the given probability P_{ed} which would be more conductive to jump out of local optimal for chemotaxis. But the event may disperse those bacteria whose position are close to the global optimum and delay the optimization process.

Particle swarm optimization algorithm: PSO algorithm is a global optimization algorithm based on swarm and fitness which originates from the movement behavior of bird swarm . The solution of each problem may be seen as a massless and no volume particle in the search pace and each particle has two features, position and velocity. The objective function value corresponding to particle coordinates can be used as the particle fitness by which the algorithm evaluates the fit and unfit quality of the individual. PSO algorithm first initializes a group of random particles and then find out the optimal solution by iteration. In each iteration, particle updates itself by tracking two extreme values: one is the best position which particle i itself has passed through, namely individual optimal solution x_{pbest_i}, the other one is the best position which the whole particle swarm has passed through, namely global optimal solution x_{gbest}. After finding the aforementioned two extreme value, particles update their velocity and position according to the following two equations:

$$v_i(t)=\omega v_i(t-1)+c_1r_1(t)(x_{pbest_i}-x_i(t))+c_2r_2(t)(x_{gbest}-x_i(t)) \quad (3)$$

$$x_i(t)=x_i(t-1)+v_i(t) \quad (4)$$

Among them, $x_i(t)$ and $v_i(t)$ represent the position and speed of the ith particle in the tth generation, ω represents non-negative inertia weight, c_1, c_2 is non-negative learning factors and r_1, r_2 is the random numbers in [0,1].

SELF-ADAPTIVE HYBRID OPTIMIZATION ALGORITHM BASED ON PSO AND BFA

Self-adaptive adjustment of learning factors in PSO: In PSO algorithm, the learning factor c_1 and c_2, respectively control the particle velocity affected by cognitive and social part. Generally speaking, in the population-based optimization method, we always hope that individuals can search in the whole optimization space in the initial stage and they are not too early to prematurely fall into local extremum while in the end stage can enhance convergence rate and accuracy and effectively find the global optimal solution. This srudy presents a kind of adjustment method that acceleration coefficient dynamically changes with the iterative process and changes based on S type function. The adjustment formula is as follows:

$$\begin{cases} c_1 = \dfrac{2}{1+\exp[a*(\dfrac{iter}{iter_{max}})-0.5]} \\ c_2 = 2-c_1 \end{cases} \quad (5)$$

In the above Eq. 5, a is a positive coefficient which controls the decreased steepness of c_1 and it is recommended to set in the interval [5,15]. $Iter_{max}$ is the maximum iterations, iter is the current iterations. The function curve of improved c_1, c_2 is shown in Fig. 1.

As shown in Fig. 1, this method can make particle have large cognitive capability to the greatest extent in the initial iterative stage while large social capability in the latter iteration stage. Thus the algorithm is more conducive to converge to global optimal solution and enhance the convergence speed and accuracy.

Self-adaptive adjustment of inertial weight in PSO: The value of inertia weight ω has important influence on the PSO optimization search. Generally, in order to obtain better algorithm performance, usually in the search early stage ω should has a greater value to ensure the particle swarm's strong global search ability within a larger search space and avoid premature. And as iterations increase, ω should has a smaller value to ensure the particle swarm's local search ability within a smaller search space and enhance the convergence precision. Therefore, the appropriate control of inertia weight in the iterative process can balance the global search and local search of algorithm, thereby getting good enough solution on average with less iteration.

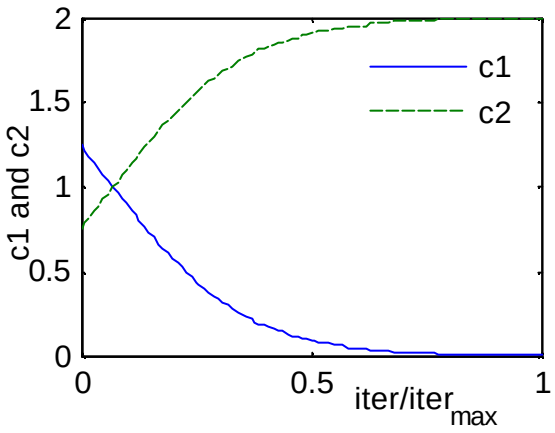

Fig. 1: Self-adaptive learning factors

Traditional self-adaptive method makes ω linearly decrease with the increase of iterations, the equation is as follows (Shi, 2001):

$$w = w_{max} - (w_{max} - w_{min}) \times (\frac{iter}{iter_{max}}) \quad (6)$$

The PSO algorithm whose ω parameter linearly decreases effectively improves the performance of algorithm but there are still some shortcomings. On the one hand, this kind of PSO algorithm can't effectively reflect the complicated non-linear behavior in the particle swarm's actual search process, so the convergence speed and convergence precision is still not ideal. On the other hand, the slope of which ω linearly decreases is still problem-dependency, there is no universal optimal change slope for all optimization problem.

By the previous analysis, the change process of ω is dynamic and non-linear, therefore, so this srudy adopts the non-linear function to describe the dynamic change rule of ω in the iteration process. The ω value of each iterative step is determined by the following exponential function equation:

$$w(iter) = w_{init} \times \exp(-(\frac{iter_{now}}{iter_{max}})^n) \quad (7)$$

In the above Eq. 7, n is a control power exponent of non-linear change rule, particularly when n = 2 Eq. 7 is often referred to as probability curve function. Figure 2 shows the ω iteration change curve with different n value.

A shown in Fig. 2., for the given initial value ω_{init} and the control power exponent n, the non-linear change rule of ω with iterations can be uniquely determined. And the greater the n value is, the longer the global search duration of particle swarm is while the smaller

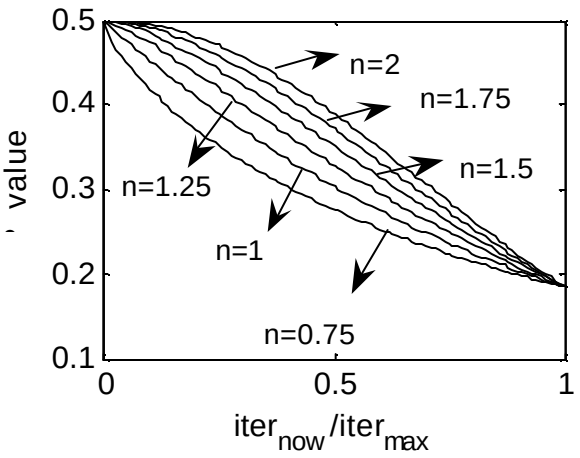

Fig. 2: ω Iteration dhange curve with different n value

the n value is, the longer the local search duration of particle swarm is. By using d-dimension spherical function:

$$f(x) = \sum_{i=1}^{d} x_i^2 \quad (8)$$

(d = 6, $x_i \in [-5.12, 5.12]$) to validate parameter values, the result shows that when ω_{init} value is set in [0.2,0.5] and n value is set in [0.5,2], the algorithm has excellent performance.

Self-adaption adjustment of chemotaxis step-length in BFA: As a very important parameter in BFA chemotactic link, the appropriate selection of chemotaxis step-length can determine the population diversity, accelerate convergence and enhance the convergence precision. Original BFA adopts fixed step-length to solve problem which is not good for the convergence of algorithm. A large number of experiments show that in the early phase the algorithm maintains a larger initial step-length for a wide range of coarse search and in the later stage maintains a smaller step-length for a small range of precision search which can not only speed up the convergence but also improve the convergence precision to a certain extent. Hence, this srudy employs the following dynamically adjusted step-length equation:

$$C(k, l) = C_{init}/m^{k+l-1} \quad (9)$$

where, C(k,l) is the chemotaxis step-length of the kth copy and the lth elimination-dispersal, C_{init} is the initial chemotaxis step-length and m is the parameter of controlling step-length's reduction gradient. C(k,l) is the function which declines with the increase of elimination-dispersal events. So when k+l is smaller, C(k,l) is larger, thereby avoiding consuming too much time in the local search range and when k+l gradually increases, C(k,l) decreases, accordingly the bacterial local search ability near the global optimal point increases and this guarantees that the algorithm eventually approaches the global optimal point.

Self-adaptive hybrid optimization algorithm based on PSO and BFA: The general idea of self-adaptive hybrid optimization algorithm based on PSO and BFA(ABSO) is under the self-adaptive adjustment strategy for learning factors and inertial weight in PSO and BFA chemokines step-length, after finishing BFA chemotaxis with variable step-size, the new algorithm continues to adjust the bacterial position with the self-adaptive PSO iteration update formula. This method can make up for the blindness of BFA random tumble, thereby helping BFA jump out of local optimal and balance the local search and global development optimization ability.

The ABSO algorithm procedure in this srudy is as follows:

- Initialize parameters D?S, Nc, Nre, Ned, Ped, Ns, C_{init}, m, a, ω_{init}, n, R_1, R_2
- Randomly initialize the bacterial position $\theta^i(1,1,1)$, i = 1, 2...S
- Calculate the initial fitness of each bacterium: $J^i(1,1,1) = Fun(\theta^i(1,1,1))$ and initialize the individual optimal value and the global optimal value of each bacterium
- For(l = 1;l<N_{ed};l++)//l is elimination-dispersal generation
- For (k = 1;k<N_{re};k++)//k is reproduction generation
- For (j = 1;j<N_c;j++)//j is chemotaxis generation
- For(i = 1;i<S;i++) //Traverse bacterial individual, i is the number of bacterial individual
- Compute the bacterial fitness, $J^i(j,k,l) = Fun(\theta^i(j,k,l))$
- Save the current optimal individual to J_{last}, $J_{last} = J^i(j,k,l)$
- Randomly tumble with self-adaptive step-size according to the ABSO Eq. 1 and 8 and update bacterial position $\theta^i(j+1,k,l)$
- Compute the bacterial fitness $J^i(j+1,k,l)$ with $\theta^i(j+1,k,l)$
- While (m_1<N_s)//m_1 is the bacterial swim step number
- If $J^i(j+1,k,l)<J_{last}$, let $J_{last} = J^i(j+1,k,l))$ and continue to swim forward with variable step-size in this same direction and then calculate the fitness-updating value
- Else terminate the swim and save the local optimal value of bacterium i to J_{last}.
- End if
- End while

- End for//End for the traverse to bacterial individual
- Assess the individual optimal position and the global optimal position in all chemotaxis operations for each bacterium
- Calculate the new velocity and position of each bacterium according to the self-adaptive PSO Eq. 3-5 and 7
- Calculate the updated fitness value of each bacterium
- End for//End for chemotaxis
- Calculate the health degree of each bacterium according to Eq. 2 and sort bacteria in order of ascending health degrees
- Split and elimination: Eliminate the latter half bacteria with poorer health degree and split the same amount of bacterial colony from the first half bacteria with better health degree
- End for//End for reproduction
- Elimination and dispersal: For the given dispersal probability P_{ed}, if a bacterium individual meets the dispersal condition, re-generate a new individual according to the random algorithm
- End for//End for elimination-dispersal

The flowchart of the ABSO algorithm is illustrated in Fig. 3.

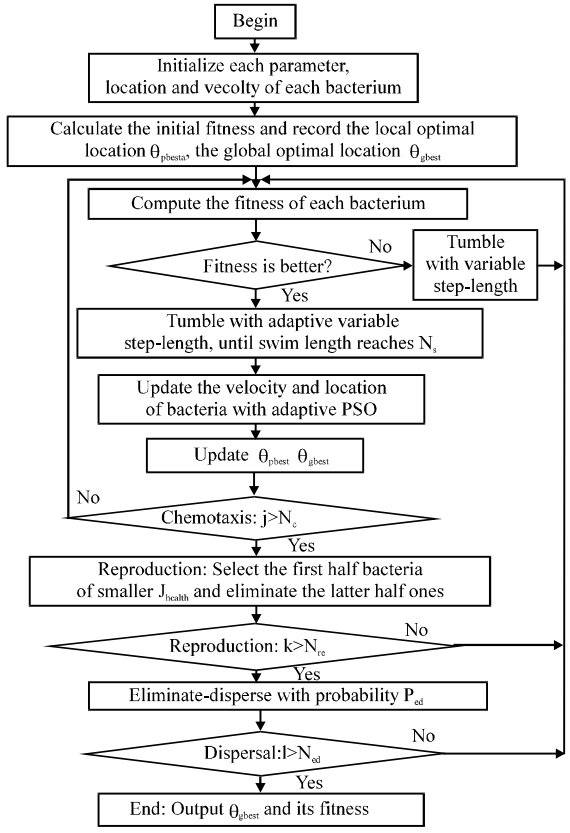

Fig. 3: Flowchart of the ABSO algorithm

SIMULATION EXPERIMENTS AND RESULTS

Test functions: In order to validate the performance of the ABSO algorithm, four commonly used Benchmark functions, shown in Table 1, are selected for numerical experiments. These functions having a large number of local optima values and with multi-peak have been widely applied to evaluate the performance of optimization algorithms and their theoretical optimal values are all 0. The number of local extreme point can surge with the increase of the test function dimension when using traditional optimization methods. Especially Rosenbrock function is recognized as morbid quadratic function difficult to minimization, whose valleys are numerous and optimization easily falls into local minimum point. Rastrigrin and Girewank functions are all typical linear multimodal functions, whose search space are all wider and have a number of local minima and tall obstructions.

Parameter settings: This paper compares the ABSO algorithm with the original BFA, PSO, self-adaptive APSO, BSOA (Biswaw *et al*., 2007; Yang *et al*., 2012), BF-PSO (Korani, 2008; Zhao *et al*., 2010). In order to guarantee the comparability of algorithms, these algorithms take the same parameters, where the population size S = 30,the variable dimension D = 30 and D = 50 respectively, the maximum dispersal generation N_{ed} = 2, the maximum chemotaxis generation N_c = 200, the maximum reproduction generation N_{re} = 5, the biggest swim steps N_s = 4, the dispersal probability P_{ed} = 0.25. We set the fixed step-length of the original BFA C = 0.002, the ABSO initial step-length C_{init} = 0.15, the control parameter of variable step-length m = 2, the control parameter of self-adaptive learning factor a = 7, the initial inertia weight ω_{init} = 0.5, the control parameter of self-adaptive inertia weight n = 1.25, the standard PSO parameters $C_1 = C_2 = 2$, ω = 0.7298, the maximum iterations $iter_{max}$ = 2000. And each experiment repeatedly runs 50 times.

Experimental results and analysis: Evaluation criteria include optimal fitness mean, standard deviation and average run time. Table 2 shows the comparison of average optima, standard deviation for six kinds of algorithms. The average optimal fitness and its standard deviation are in the first line and the second line of each table cell, respectively. We can see that the performance of BFA is the worst, the ABSO algorithm in this srudy is the best, BSOA is better than BF-PSO in the traditional mixed algorithms and APSO by using the self-adaptive thinking in this srudy is better than standard PSO in the single algorithms which also reflects the effectiveness of the self-adaptive thinking in this srudy. Whether for 30-dimensional problem or for 50-dimensional ABSO

Table 1: Test Functions

Function	Mathematical expression	Scope
Rosenbrock	$f_1(x)=\sum_{i=1}^{n-1}[100(x_{i+1}-x_i^2)^2+(x_i-1)^2]$	[-30,30]
Rastrigin	$f_2(x)=\sum_{i=1}^{n}[x_i^2-10\cos(2\pi x_i)+10]$	[-5.12,5.12]
Griewank	$f_3(x)=\frac{1}{4000}\sum_{i=1}^{n}x_i^2-\prod_{i=1}^{n}\cos(\frac{x_i}{\sqrt{i}})+1$	[-600,600]
Ackley	$f_4(x)=-20\exp(-0.2\sqrt{(\frac{1}{n}\sum_{i=1}^{n}x_i^2)})-\exp(\frac{1}{n}\sum_{i=1}^{n}\cos 2\pi x_i)+20+e$	[-32,32] parameter settings

Table 2: Average Optimal Fitness and Standard Deviation of 6 kinds of Algorithms (Mean(Standard Deviation))

Fun-ction	Dimen-sion	BFA	PSO	APSO	BSOA	BF-PSO	ABSO
f_1	30	8.283067(2.82E-2)	5.131533(7.96E-2)	5.3165(1.36E-01)	4.811333(3.48E-01)	5.230533(1.40E-01)	2.433E-1(3.10E-02)
	50	8.632667(4.51E-2)	5.869367(2.78E-2)	5.752233(3.57E-02)	5.462833(2.41E-01)	5.718667(1.42E-01)	1.49624(1.14E-01)
f_2	30	2.27 8367(.82E-2)	2.1183(2.54E-02)	1.963467(6.07E-03)	2.1127(4.73E-02)	2.5604(2.04E-02)	1.39889(6.36E-02)
	50	2. 647433(2.92E-2)	2. 5051(2.58E-02)	2. 475033(6.82E-03)	2.315767(1.13E-01)	2.232633(2.23E-02)	1.60092.79E-02)
f_3	30	2.808633(3.37E-2)	1.026867(1.25E-01)	1.003583(9.73E-02)	0.608233(1.74E-01)	1.040667(1.06E-01)	1.23E-02(2.66E-01)
	50	3.02016(5.2E-2)	1.74684(6.04E-02)	1.71828(9.97E-02)	1.32922(1.27E-01)	1.45912(9.29E-02)	1.25E-4(2.56E-01)
f_4	30	1.2792(1.69E-3)	0.987267(3.92E-02)	1.0198(2.21E-02)	0.934067(2.14E-02)	1.311567(2.76E-03)	0.7014(1.96E-03)
	50	1.295467(1.48E-3)	1.0489(2.04E-02	1.022233(2.19E-02)	1.028433(5.32E-03)	1.3146(1.50E-03)	1.0002(2.94E-03)

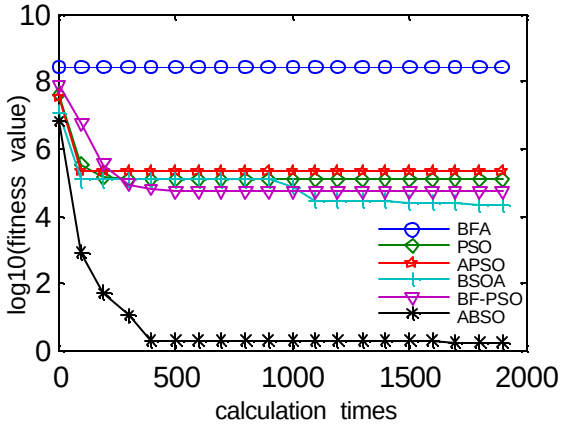

Fig. 4: Average fitness convergence curve of 30-dimension f_1 function

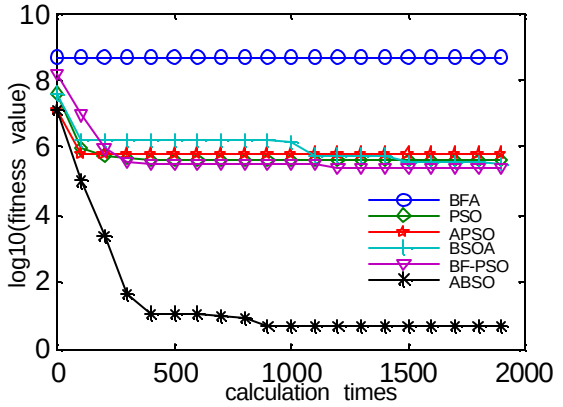

Fig. 5: Average fitness convergence curve of 50-dimension f_1 function

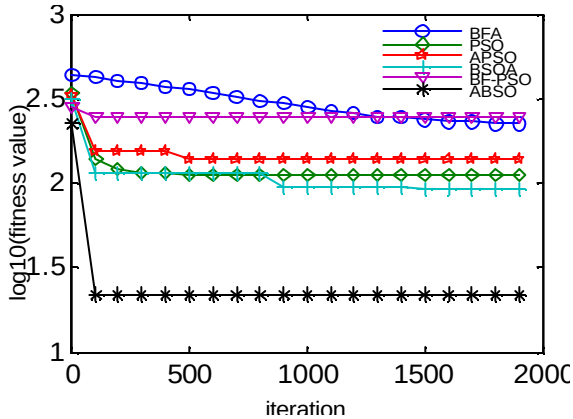

Fig. 6: Average fitness convergence curve of 30-dimension f_2 function

problem, the precision of solution is highest and standard deviation is lowest in the six algorithms. Particularly for the f_3 function, the optimal solution of ABSO is fundamentally not an order of magnitude compared with the other five algorithms. So we can see from Table 2 that ABSO has distinct superiority in accuracy and robustness which also indicates the feasibility and effectiveness of ABSO in solving the high-dimensional function problem.

Figure 4-11 show the 30-dimensional and 50-dimensional evolutionary optimization curves of four functions, whose longitudinal axises take the logarithm of the average optimal fitness of 50 times and horizontal axises represent calculation times. As can be seen from the figures, the search processes of PSO, APSO and BF-PSO are similar for f_2, f_3 functions. Under the same calculation times, the ABSO curves of four functions in

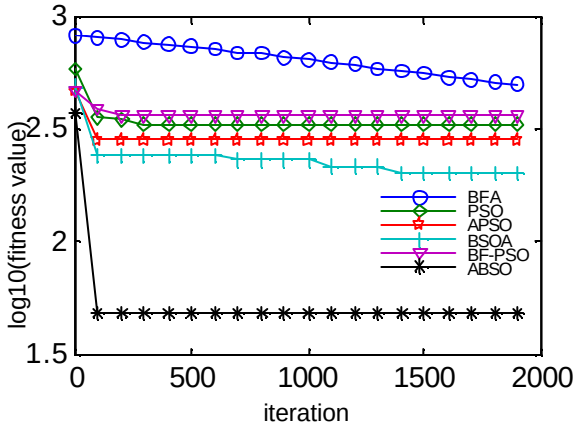

Fig. 7: Average Fitness convergence curve of 50-dimension f_2 function

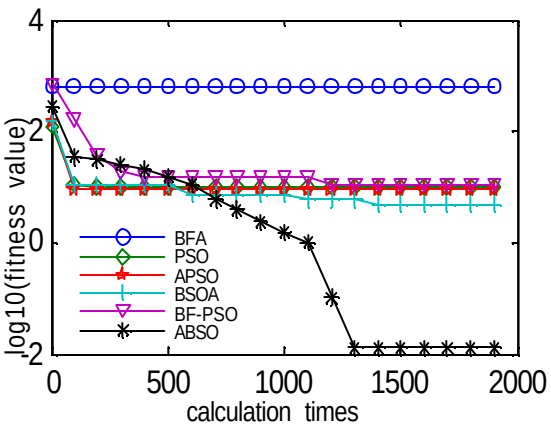

Fig. 8: Average fitness convergence curve of 30-dimension f_3 function

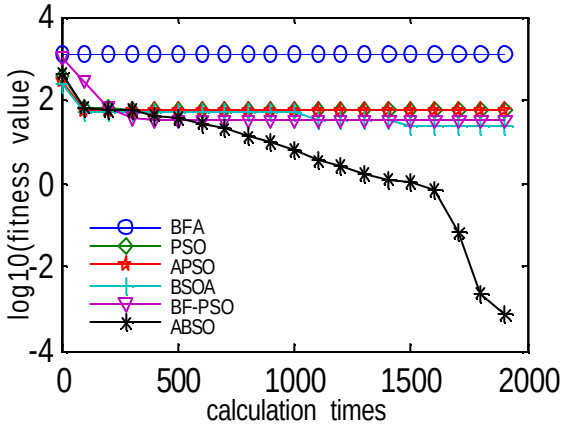

Fig. 9: Average fitness convergence curve of 50-dimension f_3 function

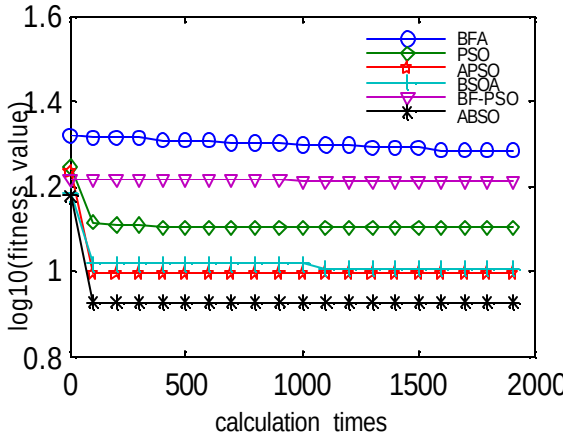

Fig. 10: Average fitness convergence curve of 30-dimension f_4 function

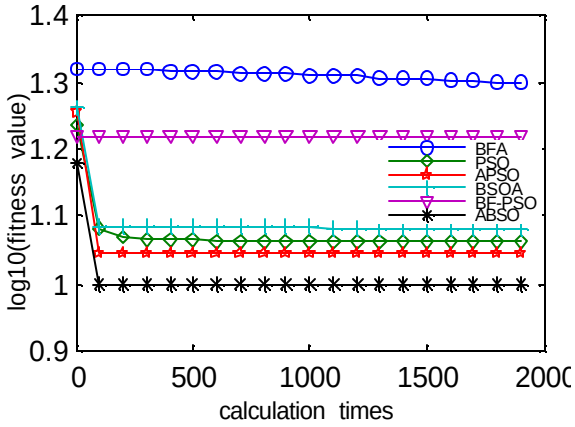

Fig. 11: Average fitness convergence curve of 50-dimension f_4 function

the six algorithms decline most quickly and have the fastest convergence speed which is the most apparent in convergence curve of f_1, f_3 functions. By the comparisons for two kinds of dimensions in 6 kinds of algorithms, we find that the ability of approximating optimal solution of the 50-dimensional ABSO algorithm is greatest, especially for the 50-dimensional f_3 function, its convergence phenomenon is most prominent. This shows that the higher the algorithm complexity is, the faster the ABSO convergence speed is.

Table 3 shows the average run time of six algorithms. As can be seen from the table, the ABSO algorithm is a little slower than PSO and APSO due to the combination of BFA characteristics but ABSO is much faster than BFA and the other kinds hybrid algorithms including BSOA and BF-PSO.

Table 3: Average run time of 6 kinds of algorithms(/s)

Fun-ction	Dimen-sion	BFA	PSO	APSO	BSOA	BF-PSO	ABSO
f_1	30	10.5833	2.24466	2.219	5.92667	13.8857	5.078
	50	11.146	2.43766	2.40633	6.23433	17.752	5.682
f_2	30	12.0136	3.1803	3.092	6.82533	8.15633	5.08867
	50	14.4376	4.18733	3.91133	7.943	9.62467	6.873
f_3	30	14.2967	4.30731	4.138	7.4635	22.227	8.3625
	50	17.38	5.744	5.434	8.8034	26.403	16.288
f_4	30	12.6877	3.47933	3.33333	6.81233	8.34366	5.69267
	50	14.7763	4.42166	4.19767	7.667	9.35933	7.10967

The above synthesis results indicate that ABSO has higher search precision, faster convergence speed, greater global convergence ability and robustness.

CONCLUSIONS

This study presented a self-adaptive hybrid optimization algorithm based on PSO and BFA, namely ABSO. The new algorithm combined BFA and PSO effectively, not only maintained the capability of PSO's fast convergence and that of BFA's obtaining global optimal solution but also made up for the defects of BFA's easily trapped in local optimum with the PSO update idea. While the self-adaption of parameters further enhanced the local search ability of the ABSO algorithm. Simulation results show that the new algorithm is superior to exsiting algorithms in the optimal solution accuracy, convergence speed and robustness which proves that the validity of the improved algorithm and the algorithm is more suitable for solving the optimization problems of complex functions with high-dimension. The subsequent research will further focus on the parameter settings and search direction, apply the improved algorithm for discrete and multi-objective optimization problem and ultimately for engineering practice.

ACKNOWLEDGMENTS

We would like to thank to the reviewers for their helpful comments. This work was financially supported by the Natural Science Foundation of China (#1072166), the Higher School Science and Technology Development Project in Shanxi Province of China (#2013147) and the key discipline construction project of Xinzhou Teachers' University (# ZDXK201204).

REFERENCES

Bakwad, K.M., S.S. Pattnaik, B.S. Sohi, S. Devi, B.K. Panigrahi, S. Das and M.R. Lohokare, 2009. Hybrid bacterial foraging with parameter free PSO. Proceedings of the World Congress on Nature and Biologically Inspired Computing, December 9-11, 2009, Coimbatore, pp: 1077-1081.

Biswas, A., S. Dasgupta, S. Das and A. Abraham, 2007. Synergy of PSO and Bacterial Foraging Optimization-a Comparative Study on Numerical Benchmarks. In: Innovations in Hybrid Intelligent Systems, Corchado, E., J.M. Corchado and A. Abraham (Eds.). Springer, Berlin Heidelberg, pp: 255-263.

Chen, H., Y. Zhu and K. Hu, 2008. Self-adaptation in bacterial foraging optimization algorithm. Proceedings of the 3rd International Conference on Intelligent System and Knowledge Engineering, Volume 1, November 17-19, 2008, Xiamen, pp: 1026-1031.

Chu, Y., H. Mi, H. Liao, Z. Ji and Q.H. Wu, 2008. A fast bacterial swarming algorithm for high-dimensional function optimization. Proceedings of the IEEE Congress on Evolutionary Computation, June 1-6, 2008, Hong Kong, pp: 3135-3140.

Biswas, A., S. Dasgupta, S. Das and A. Abrahamy, 2007. A synergy of differential evolution and bacterial foraging optimization for global optimization. Neural Network World, 17: 607-626.

Dasgupta, S., S. Das, A. Abraham and A. Biswas, 2009. Adaptive computational chemotaxis in bacterial foraging optimization: An analysis. IEEE Trans. Evol. Comput., 13: 919-941.

Datta, T., I.S. Misra, B.B. Mangaraj and S. Imtiaj, 2008. Improved adaptive bacteria foraging algorithm in optimization of antenna array for faster convergence. Prog. Electromag. Res., C, 1: 143-157.

Korani, W.M., 2008. Bacterial foraging oriented by particle swarm optimization strategy for PID tuning. Proceedings of the GECCO Conference Companion on Genetic and Evolutionary Computation, July 12-16, 2008, Atlanta, Georgia, USA, pp: 1823-1826.

Majhi, R., G. Panda, B. Majhi and G. Sahoo, 2009. Efficient prediction of stock market indices using adaptive Bacterial Foraging Optimization (ABFO) and BFO based techniques. Expert Syst. Appl., 36: 10097-10104.

Passino, K.M., 2002. Biomimicry of bacterial foraging for distributed optimization and control. IEEE Control Syst., 22: 52-67.

Shi, Y. and R.C. Eberhart, 2001. Fuzzy self-adaptive particle swarm optimization. Proceedings of the IEEE Congress on Evolutionary Computaiton, May 27-30, 2001, IEEE Service Center, Piscataway, New Jersey, pp: 101-106.

Tang, W.J., Q.H. Wu and J.R. Saunders, 2007. A bacterial swarming algorithm for global optimization. Proceedings of the Congress on Evolutionary Computation, September 25-28, 2007, Singapore, pp: 1207-1212.

Yang, S.J., S.W. Wang, J. Tao and X. Liu, 2012. Multi-objective optimization method based on hybrid swarm intelligence algorithm. Comput. Simul., 29: 218-222.

Zhao, Q., Z. Wu and G. Meng, 2010. A new hybrid intelligent algorithm and its application on attribute reduction. J. Wuhan Univ. (Nat. Sci. Edn.), 56: 723-728.