Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2014, Article ID 481791, 9 pages
http://dx.doi.org/10.1155/2014/481791

Research Article

Flower Pollination Algorithm with Dimension by Dimension Improvement

Rui Wang and Yongquan Zhou

College of Information Science and Engineering, Guangxi University for Nationalities, Nanning 530006, China

Correspondence should be addressed to Yongquan Zhou; yongquanzhou@126.com

Received 8 July 2014; Accepted 16 August 2014; Published 1 September 2014

Academic Editor: Shifei Ding

Flower pollination algorithm (FPA) is a new nature-inspired intelligent algorithm which uses the whole update and evaluation strategy on solutions. For solving multidimension function optimization problems, this strategy may deteriorate the convergence speed and the quality of solution of algorithm due to interference phenomena among dimensions. To overcome this shortage, in this paper a dimension by dimension improvement based flower pollination algorithm is proposed. In the progress of iteration of improved algorithm, a dimension by dimension based update and evaluation strategy on solutions is used. And, in order to enhance the local searching ability, local neighborhood search strategy is also applied in this improved algorithm. The simulation experiments show that the proposed strategies can improve the convergence speed and the quality of solutions effectively.

1. Introduction

In recent years, more and more bioinspired algorithms are proposed, such as genetic algorithm (GA) [1], simulated annealing (SA) [2], particle swarm optimization (PSO) [3], firefly algorithm (FA) [4], glowworm swarm optimization (GSO) [5], monkey search (MS) [6], bacterial foraging optimization algorithm (BFOA) [7], invasive weed optimization (IWO) [8], cultural algorithms (CA) [9], and harmony search (HS) [10]. Because of their advantages of global and parallel efficiency, robustness, and universality, swarm intelligence algorithms have been widely used in engineering optimization, scientific computing, automatic control, and other fields.

Flower pollination algorithm (proposed by Yang in 2012) [11] is a new population-based intelligent optimization algorithm by simulating flower pollination behavior. And FPA has been extensively researched to solve Integer Programming Problems [12], Sudoku Puzzles [13], and Wireless Sensor Network Lifetime Global Optimization [14] in the last two years by scholars. It is estimated that there are over 250,000 types of flowering plants in nature. And researchers of biology considered that almost four-fifths of all plant species are flowering species. Flower pollination behavior stems from the purpose of reproduction. From the biological evolution point of view, the objective of flower pollination is the survival of the fittest and the optimal reproduction of species. All these factors and processes of flower pollination interact so as to achieve optimal reproduction of the flowering plants. In nature, pollination can be divided into two parts: abiotic and biotic. Almost 90% pollen grains are transferred by insects and animals; we call this biotic pollination. The other 10% pollen grains are transferred by wind [15, 16]. They do not need pollinators. And we call this form abiotic pollination. Pollinators can be very diverse; researches show almost 200,000 kinds of pollinators.

Self-pollination and cross-pollination are two different ways of pollination [17]. Cross-pollination means pollination can occur from pollen of a flower of a different plant, and self-pollination is just the opposite. Biotic, cross-pollination can occur at long distance; the pollinators such as bees, bats, birds can fly a long distance; thus they can be considered as the global pollinators. And these pollinators can fly as Lévy flight behavior [18], with fly distance steps obeying a Lévy distribution. Thus, this can inspire to design new optimization algorithm. Flower pollination algorithm is an optimization algorithm which simulates the flower pollination behavior mentioned above; flower pollination algorithm can also be divided into global pollination process and local pollination process.

2. FPA with Dimension by Dimension Improvement

In order to enhance the global searching and local searching abilities, we applied three optimization strategies to basic flower pollination algorithm (FPA); those were local neighborhood searching strategy (LNSS) [19], dimension by dimension evaluation and improvement strategy (DDEIS), and dynamic switching probability strategy (DSPS).

2.1. Local Neighborhood Search Strategy (LNSS). FPA (developed by Yang and Deb) uses differential evolution (DE) algorithm [20] to do local search. And experiment results show that the local search ability of DE is limited. Thus, we add LNSS to local search process to enhance its exploitation ability.

Firstly, we should explain a model (local neighborhood model). In this model, each vector uses the best vector of only a small neighborhood rather than the entire population to do the mutation. We suppose that there exists a differential evolutionary population $P_G = [X_{1,G}, X_{2,G}, \ldots, X_{i+1,G}]$, and each $X_{i,G}$ $(i = 1, 2, 3, \ldots, NP)$ is a parameter vector, and its dimension is D. Each vector subscript index is randomly divided to ensure the diversity of each neighborhood. For each vector $X_{i,G}$, we can define a neighborhood, and the radius is k $(2k + 1 < NP)$. The neighborhood consists of vector $X_{i-k,G}, \ldots, X_{i,G}, \ldots, X_{i+k,G}$. Assume that the vectors accord the subscript indices in a ring topology structure. We can take $X_{NP,G}$ and $X_{2,G}$ as two direct neighbors of $X_{1,G}$. The concept of local neighborhood model is shown in Figures 1 and 2. The neighborhood topology here is static and determined by the collection of vector subscript indices. And the local neighborhood model can be expressed in the following formula:

$$X_{i,G+1} = X_{i,G} + \alpha \left(X_{n\_\text{best}_i,G} - X_{i,G} \right) + \beta \left(X_{p,G} - X_{q,G} \right), \tag{1}$$

where $X_{n\_\text{best}_i,G}$ is the best vector of $X_{i,G}$ neighborhood, $p, q \in [i - k, i + k]$ $(p \neq q \neq i)$, and α, β are two scale factors.

2.2. Dimension by Dimension Evaluation and Improvement Strategy (DDEIS). Flower pollination algorithm uses the whole update and evaluation strategy on solutions. For solving multidimensional function optimization problems, this strategy may deteriorate the convergence speed and the quality of solution due to interference phenomena among dimensions. To overcome this shortage, we add this strategy to FPA in local search process.

In FPA, Lévy flight can improve the diversity of population and strengthen the global search ability of the algorithm. But for multidimensional objective function, overall update evaluation strategy will affect the convergence rate and quality of solutions. DDEIS updates dimension by dimension.

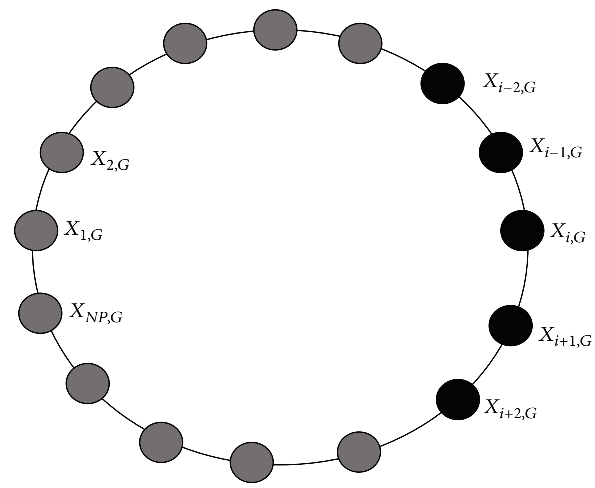

Figure 1: Neighborhood ring topology.

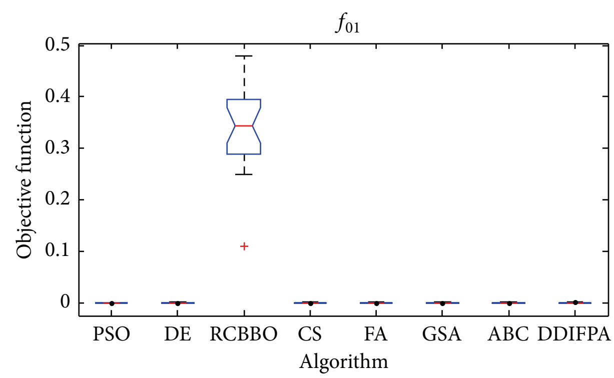

Figure 2: ANOVA tests for f_{01}.

Assume that objective function is $f(X) = X_1^2 + X_2^2 + X_3^2$ and $X_{g,i} = (0.5, 0.5, 0.5)$ is a solution to $f(X)$.

The objective value $f(X_{g,i}) = 0.75$. We use formula (1) to update $X_{g,i}$ and get $X_{g+1,i} = (0, 1, -1)$. For example, when first dimension value of $X_{g,i}$ updates from 0.5 to 0, combined with the value of other dimensions, we can get a new $X_{g+1,i} = (0, 0.5, 0.5)$. The objective value $f(X_{g+1,i}) = 0.5 < f(X_{g,i})$; it can improve current solution. Thus, we accept this update and update operation into the next dimension. If first dimension value of $X_{g,i}$ updates from 0.5 to 1, we can get a new $X_{g+1,i} = (1, 0.5, 0.5)$. The objective value $f(X_{g+1,i}) = 1.5 > f(X_{g,i})$, it fails to improve $X_{g,i}$, and we should abandon the current dimension updated value and update operation into the next dimension. The strategy is described in Algorithm 1.

2.3. Dynamic Switching Probability Strategy (DSPS). In FPA, local search and global search are controlled by a switching probability $p \in [0, 1]$, and it is a constant value. We suppose that a reasonable algorithm should do more global search at the beginning of searching process and global search should be less in the end. Thus, we applied the dynamic switching probability strategy (DSPS) to adjust the proportion of two

```
temp2 = X_{g,i};
temp = X_{g+1,i};
for m = 1 : d
    temp3 = temp2;
    temp2(m) = temp(m);
    if fitness(temp2 ) > fitness(temp3),
        temp2(m) = X_{g,i}(m);
    endif
endfor
```

Algorithm 1: Dimension by dimension evaluation and improvement strategy.

kinds of searching process. Switching probability p can alter according to the following formula:

$$p = 0.6 - 0.1 \times \frac{(\text{Max\_iter} - t)}{\text{Max\_iter}}, \tag{2}$$

where Max_iter is the maximum iterations of the DDIFPA and t is current iteration. Specific implementation steps of FPA with dimension by dimension improvement (DDIFPA) can be summarized in the pseudocode shown in Algorithm 2.

3. Numerical Simulation Experiments

In this section, we applied 12 standard test functions [21] to evaluate the optimal performance of FPA with dimension by dimension improvement (DDIFPA). The mean and standard deviation results of 20 independent runs for each algorithm have been summarized in Table 2. The 12 standard benchmark functions have been widely used in the literature. The dimensions, scopes, optimal values, and iterations of 12 functions are in Table 1. We also do some high-dimensional tests, and the results are showed in Table 3.

3.1. Experimental Setup. All of the algorithm was programmed in MATLAB R2012a; numerical experiment was set up on AMD Athlont (tm) II∗4640 processor and 2 GB memory.

3.2. Comparison of Each Algorithm Performance. The proposed DDIFPA algorithm is compared with mainstream swarm intelligence algorithms FPA [11], PSO [22], DE [23], RCBBO [24], GSA [25], FA [26], CS [27], and ABC [28], respectively, using the mean and standard deviations to compare their optimal performances. The setting values of algorithm control parameters of the mentioned algorithms are given as follows.

PSO parameters setting: weight factor $\omega = 0.6$, $c_1 = c_2 = 2$. The population size is 100 [22].

DE parameters setting: $F = 0.5$ and $CR = 0.9$ in accordance with the suggestions given in [23]; the population size is 100.

ABC parameters setting: *limit* $= 5D$ has been used as recommended in [22]; the population size is 50 because this algorithm has two phases.

RCBBC parameters setting: maximum immigration rate: $I = 1$, maximum emigration rate: $E = 1$, and *mutation probability*: $m_{\max} = 0.005$ have been used as recommended in [24]; the population size is 100.

CS parameters setting: $\beta = 1.5$ and $\rho_0 = 1.5$ have been used as recommended in [27]; the population size is 50 because this algorithm has two phases.

GSA parameters setting: $G_0 = 100, \alpha = 20$ and K_0 which is set to NP and is decreased linearly to 1 have been used as recommended in [25]; the population size is 100.

FA parameters setting: $\alpha_0 = 0.5$, $\beta_0 = 0.2$, and $\gamma = 1$ have been used as recommended in [26]; the population size is 100.

FPA parameters setting: the population size is 50 because this algorithm has two phases [11].

DDIFPA parameters setting: the population size is 50 because this algorithm has two phases.

From the rank of each function in Table 2, we can conclude that DDIFPA provides many of the best results are better than FPA and other algorithms, especially for functions f_{01}, f_{03}, and f_{05}. For f_{01} the mean and standard deviation of DDIFPA are much higher than FPA. For f_{03}, the mean and standard deviation of DDIFPA are 117 orders of magnitude higher than GSA and 127 orders of magnitude higher than FPA. For f_{04}, DDIFPA and FPA fail to give the best optimal solution. For f_{05}, the mean and standard deviation of DDIFPA are 2 orders of magnitude higher than FPA.

Figures 2 and 3 show the graphical analysis results of ANOVA test. As can be seen in Figure 2, when solving function f_{01}, most of the algorithms can obtain the stable optimal value after 20 independent runs except RCBBO algorithm, and, in Figure 3, when solving the function f_{05}, DDIFPA is more stable than other algorithms.

Figures 4 and 5 show the fitness function curve evolution of each algorithm for f_{01} and f_{05}. From the two figures, we can conclude that DDIFPA has a faster convergence rate and a higher optimizing precision.

For multimodal functions f_{06} to f_{10} with many local minima, the final results are more important because these functions can reflect the ability of algorithm to escape from poor local optima and obtain the global optimum.

As can be seen in Table 2, for f_{06} and f_{07}, DDIFPA are in first place; ABC achieve the optimal value when solving f_{07}. For f_{08} the mean and standard deviation of DDIFPA are 15 orders of magnitude higher than FPA. For f_{09}, ABC and DDIFPA all achieve the optimal value and the standard

Objective min or max $f(x), x = (x_1, x_2, \ldots, x_d)$
Initialize a population of n flowers/pollen gametes with random solutions
Find the best solution g_* in the initial population
while ($t < MaxGeneration$)
  for $i = 1 : n$ (all n flowers in the population)
    Get p according to formula (2);
    if rand $< p$
      Draw a (d-dimensional) step vector L which obeys a Lévy distribution
      Global pollination via $x_i^{t+1} = x_i^t + \gamma L(\lambda)(g_* - x_i^t)$;
    else
      Draw ε from a uniform distribution in [0, 1];
      Local pollination via $X_{i,G+1} = X_{i,G} + \alpha(X_{n\_best_i,G} - X_{i,G}) + \beta(X_{p,G} - X_{q,G})$;
      where $\alpha = \beta = \varepsilon$;
    end if
      Evaluate new solutions via DDEIS
      If new solutions are better, update them in the population
  end for
      find the current best solution g_*
end while

Algorithm 2: FPA with dimension by dimension improvement (DDIFPA).

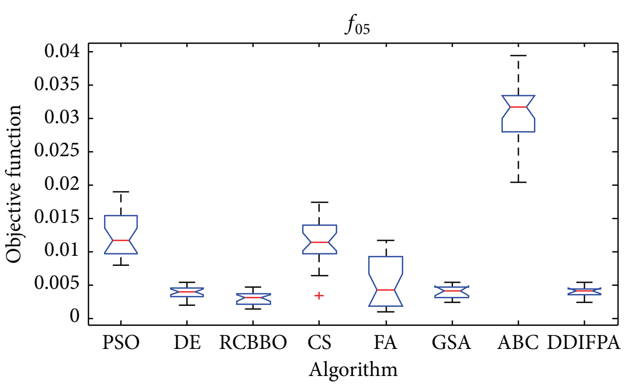

Figure 3: ANOVA tests for f_{05}.

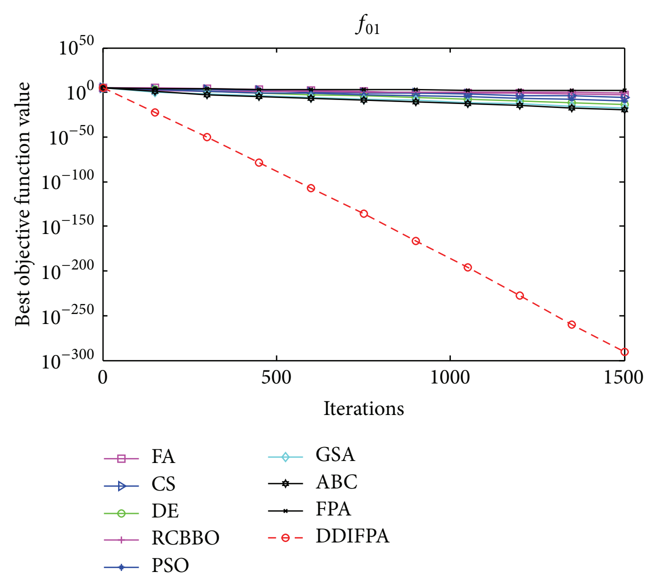

Figure 4: Fitness function curve evolution for f_{01}.

deviations are all 0. For f_{10}, the mean of DDIFPA is 11 orders of magnitude higher than ABC, and the standard deviation of DDIFPA is 27 orders of magnitude higher than ABC.

Figures 6 and 7 show the graphical analysis results of the ANOVA tests. Figure 6 shows that RCBBO, ABC, and DDIFPA can obtain the relatively stable optimal values. Figure 7 shows that when solving function f_{10}, most of the algorithms can obtain the stable optimal value after 20 independent runs.

Figures 8 and 9 show the fitness function curve evolution. From Figure 9, we can conclude that both ABC and DDIFPA converge to the optimal solution. From Figure 9, we can conclude that DDIFPA converges to a more precise point than other algorithms, and its convergence speed is faster.

From Table 2, f_{11} and f_{12} are multimodal low-dimensional functions. For f_{11}, the solutions of most of the algorithms are accurate in 3 to 4 decimal places, and the rank of DDIFPA is second. For f_{12}, the rank is second too, and the experiment results show that DDIFPA can do a good job in solving multimodal low-dimensional problems.

3.3. Experimental Analysis. We have carried out benchmark validations for unimodal and multimodal test functions using the proposed algorithm (DDIFPA) with three improvement strategies (local neighborhood search strategy, dimension by dimension evaluation and improvement strategy, and dynamic switching probability strategy). An optimization

Table 1: Benchmark test functions.

Benchmark test functions	Dimension	Range	Optimum	Iterations
$f_{01} = \sum_{i=1}^{n} x_i^2$	30	$[-100, 100]$	0	1500
$f_{02} = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	30	$[-10, 10]$	0	2000
$f_{03} = \max_i \{\lvert x_i \rvert, 1 \leq i \leq D\}$	30	$[-100, 100]$	0	5000
$f_{04} = \sum_{i=1}^{D-1} \left[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2\right]$	30	$[-30, 30]$	0	5000
$f_{05} = \sum_{i=1}^{D} i x_i^4 + \text{random}[0, 1)$	30	$[-1.28, 1.28]$	0	3000
$f_{06} = \sum_{i=1}^{D} - x_i \sin\left(\sqrt{\lvert x_i \rvert}\right)$	30	$[-500, 500]$	$-418.9829^{*}n$	3000
$f_{07} = \sum_{i=1}^{D} \left[x_i^2 - 10\cos(2\pi x_i) + 10\right]$	30	$[-5.12, 5.12]$	0	3000
$f_{08} = -20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos 2\pi x_i\right) + 20 + e$	30	$[-32, 32]$	0	1500
$f_{09}(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{2}}\right) + 1$	30	$[-600, 600]$	0	2000
$f_{10}(x) = \frac{\pi}{D}\left\{\begin{array}{c} 10\sin^2(\pi y_1) \\ +\sum_{i=1}^{D-1}(y-1)^2\left[1 + 10\sin^2(\pi y_1)\right] + (y_D - 1)^2 \end{array}\right\} + \sum_{i=1}^{D} u(x_i, 10, 100, 4)$, $y_i = 1 + \frac{x_i + 1}{4}$, $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m, x_i > a \\ 0, -a \leq x_i \leq a \\ k(-x_i - z)^m, x_i < a \end{cases}$	30	$[-50, 50]$	0	1500
$f_{11}(x) = -\sum_{i=1}^{4} c_i \exp\left[\sum_{j=1}^{3} a_{ij}\left(x_j - p_{ij}\right)^2\right]$	3	$[0, 1]$	-3.8628	100
$f_{12}(x) = -\sum_{i=1}^{10}\left[(X - a_i)(X - a_i)^T + c_i\right]^{-1}$	4	$[0, 10]$	-10.5364	100

process can be divided into two key components (local search and global search); we use a dynamic switching probability $p \in [0, 1]$ to control the whole searching process. LNSS and DDEIS are applied to the local search process and enhance its exploitation ability. Among 12 test functions listed above, f_{01} to f_{05} are unimodal, and the remarkable achievements confirm that DDIFPA have stronger exploitation ability than FPA and other algorithms. And DSPS, which could improve the ability of escape from poor local optima, was applied to enhance the exploration ability. That also balanced exploitation and exploration dynamically. For multimodal benchmark functions (f_{06} to f_{12}), we can conclude that DDIFPA converges to a more precise point than other algorithms, and its convergence speed is faster. Our simulation results for finding the global optima of various test functions suggest that DDIFPA can outperform the FPA and other mentioned algorithms in terms of both precision and convergence speed.

3.4. High-Dimensional Functions Test. In previous sections, 12 standard test functions are applied to evaluate the optimal performances of the FPA with dimension by dimension improvement (DDIFPA) in the case of low dimension. In order to evaluate the performances of DDIFPA comprehensively, we also do some high-dimensional tests in $f_1, f_2, f_4, f_7, f_{10}$. The test results are shown in Table 3. As can be seen in Table 3, DDIFPA can also solve high-dimensional problems efficiently and stably.

4. Conclusions

In this paper, three optimization strategies (local neighborhood search strategy, dimension by dimension evaluation and improvement strategy, and dynamic switching probability strategy) have been applied to FPA to improve its deficiencies. By 12 typical standard benchmark functions simulation,

Table 2: Experiment results of bench mark functions for different algorithms.

Functions		PSO	DE	RCBBO	CS	FA	GSA	ABC	FPA	**DDIFPA**
f_{01}	Mean	3.33E − 10	5.60E − 14	0.3737	5.66E − 06	1.70E − 03	3.37E − 18	2.99E − 20	123.5791661	4.62E − 289
	Std.	7.04E − 10	4.41E − 14	0.1181	2.86E − 06	4.06E − 04	8.09E − 19	2.15E − 20	52.5878636	0
	Rank	5	4	8	6	7	3	2	9	1
f_{02}	Mean	6.66E − 11	4.73E − 10	0.1656	2.00E − 03	4.53E − 02	8.92E − 09	1.42E − 15	8.27840887	3.61E − 196
	Std.	9.26E − 11	1.78E − 10	0.0342	8.10E − 04	3.38E − 02	1.33E − 09	5.53E − 16	2.19245633	0
	Rank	3	4	8	6	7	5	2	9	1
f_{03}	Mean	7.9997	0.2216	7.9738	3.2388	0.0554	9.93E − 10	18.5227	3.759984116	4.52E − 127
	Std.	2.535	0.243	2.6633	0.6644	0.0101	1.19E − 10	4.2477	1.301640714	5.83E − 127
	Rank	7	4	8	5	3	2	9	6	1
f_{04}	Mean	46.9202	0.2657	64.6907	8.0092	38.1248	20.0819	0.0441	32.25150881	0.065392617
	Std.	38.0312	1.0293	36.2782	1.9188	30.3962	0.1722	0.0707	12.70746052	0.094325113
	Rank	8	3	9	4.0000	7	5	1	6	2
f_{05}	Mean	0.0135	0.0042	0.003	0.0096	0.0082	0.0039	0.0324	0.029755135	0.003729368
	Std.	0.0041	0.0014	0.0012	0.0028	0.0093	0.0013	0.0059	0.013355902	0.000996113
	Rank	7	4	1	6	5	3	9	8	2
f_{06}	Mean	−8.83E + 03	−1.13E + 04	−1.26E + 04	−9.15E + 03	−6.22E + 03	−3.05E + 03	−1.25E + 04	−8448.868832	−12569.48662
	Std.	611.159	1.81E + 03	0.5758	2.53E + 02	7.72E + 02	3.39E + 02	61.1186	292.6519355	1.92E − 12
	Rank	6	4	2	5	8	9	3	7	1
f_{07}	Mean	18.2675	134.6789	0.0385	51.2202	23.5213	7.2831	0	75.97229041	0
	Std.	4.7965	28.8598	0.0154	8.1069	8.3683	1.8991	0	12.38031696	0
	Rank	5	9	3	7	6	3	1	8	1
f_{08}	Mean	3.87E − 06	7.47E − 08	0.1947	2.375	0.0094	1.47E − 09	1.19E − 09	3.706074906	4.44E − 15
	Std.	2.86E − 06	3.11E − 08	0.0461	1.1238	0.0014	1.44E − 10	5.01E − 10	0.519323445	0
	Rank	5	4	7	8	6	2	3	9	1
f_{09}	Mean	0.0168	0	0.2765	4.49E − 05	0.0025	0.01265	0	4.559765038	0
	Std.	0.0205	0	0.0796	8.96E − 05	4.69E − 04	0.0216	0	1.769274137	0
	Rank	7	1	8	4	5	6	1	9	1
f_{10}	Mean	0.0083	4.71E − 15	0.002	0.5071	8.87E − 06	2.04E − 20	1.19E − 21	4.3551	1.57E − 32
	Std.	0.0287	3.26E − 15	0.0023	0.2662	2.80E − 06	4.53E − 21	1.08E − 21	1.1793	2.89E − 48
	Rank	7	4	6	8	5	3	2	9	1
f_{11}	Mean	−3.8628	−3.8628	−3.8627	−3.8628	−3.8613	−3.8625	−3.8628	−3.861091803	−3.862782148
	Std.	3.13E − 12	2.30E − 15	1.41E − 04	1.40E − 05	0.0037	3.88E − 04	1.37E − 10	0.00149557	1.43E − 13
	Rank	3	1	6	5	8	7	4	9	2
f_{12}	Mean	−8.9611	−10.5364	−9.3514	−9.7534	−10.2297	−8.2651	−10.5339	−5.006727392	−10.53636839
	Std.	2.8381	3.97E − 06	2.6288	0.4913	1.5332	2.8868	0.0054	1.235737304	7.4406E − 05
	Rank	7	1	6	5	4	8	3	9	2

Table 3: High-dimensional functions test results.

Functions	Dimensions	Means	Std.	Best	Worst
f_{01}	1000	$6.47933690759837E-284$	0	$9.49725772050939E-288$	$3.19181958962329E-283$
f_{02}	500	$2.24079551697238E-193$	0	$5.54622672996093E-195$	$5.86867756931104E-193$
f_{04}	500	$3.4272355502299E-17$	$2.42530519040223E-17$	$6.91784434902547E-18$	$6.99285915411276E-17$
f_{07}	500	0	0	0	0
f_{10}	500	$1.57054477178664E-32$	0	$1.57054477178664E-32$	$1.57054477178664E-32$

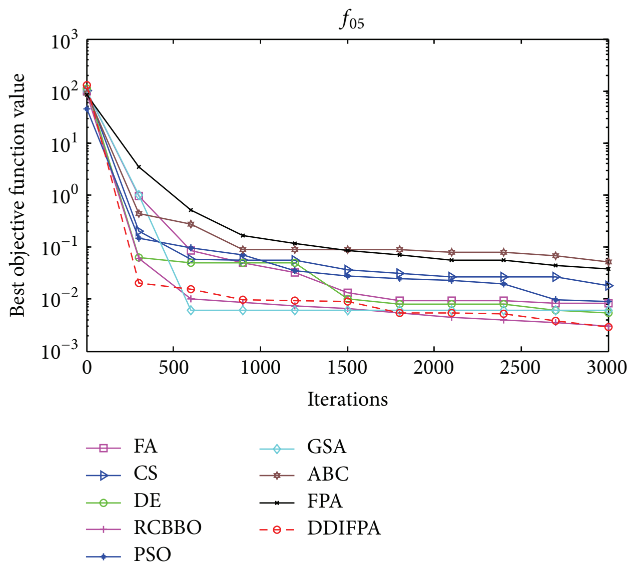

Figure 5: Fitness function curve evolution for f_{05}.

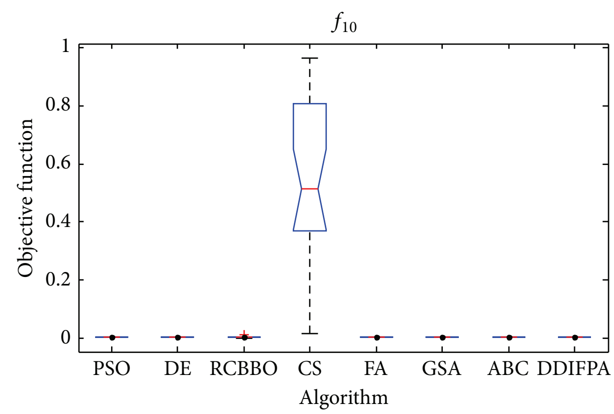

Figure 7: ANOVA tests for f_{10}.

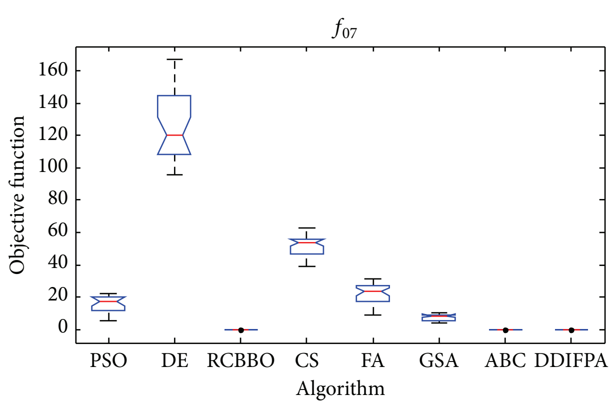

Figure 6: ANOVA tests for f_{07}.

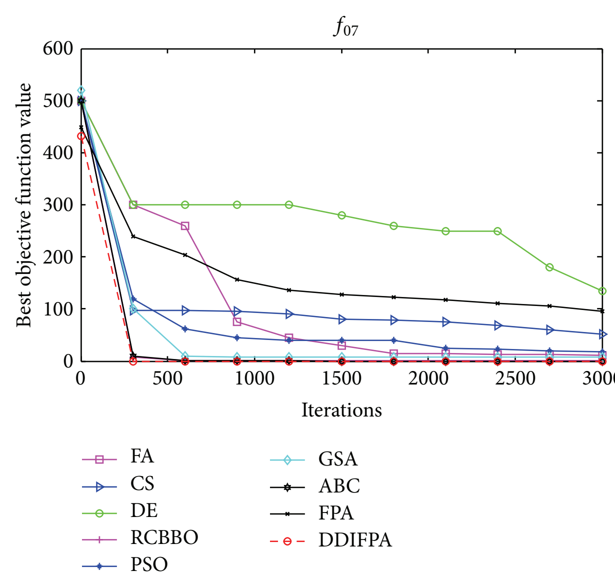

Figure 8: Fitness function curve evolution for f_{07}.

the results show that DDIFPA algorithm generally has strong global searching ability and local optimization ability, and effectively avoid the defects of other algorithms fall into local optimization. DDIFPA has improved the convergence speed and convergence precision of FPA. The experiment results show that it is an effective algorithm to solve complex functions optimization problems.

In this paper, we only consider the global optimization. The algorithm can be extended to solve other problems such as constrained optimization problems and multiobjective optimization problem. In addition, many engineering design problems are typically difficult to solve. The application of the proposed FPA with dimension by dimension improvement in engineering design optimization may prove fruitful.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

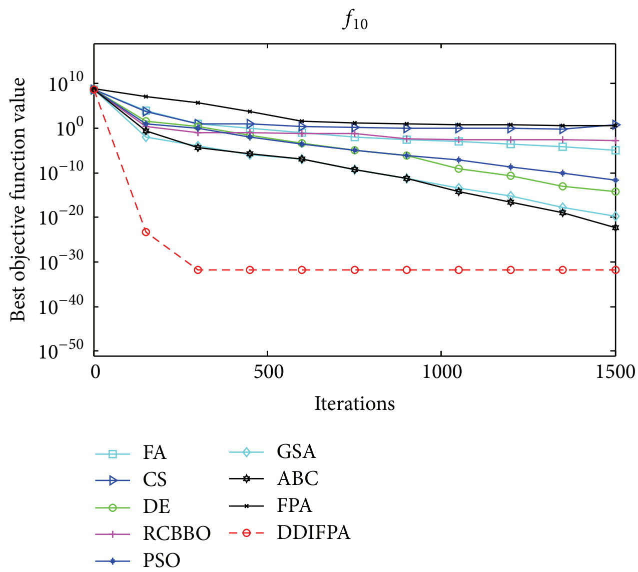

FIGURE 9: Fitness function curve evolution for f_{10}.

Acknowledgments

This work is supported by the National Science Foundation of China under Grant nos. 61165015 and 61463007, the Key Project of Guangxi Science Foundation under Grant no. 2012GXNSFDA053028, and the Key Project of Guangxi High School Science Foundation under Grant nos. 20121ZD008 and 201203YB072.

References

[1] J. H. Holland, *Adaptation in Natural and Artificial Systems*, MIT Press, Cambridge, Mass, USA, 1992.

[2] S. Kirkpatrick Jr., C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 11, pp. 650–671, 1983.

[3] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, vol. 4, pp. 1942–1948, Perth, Australia, 1995.

[4] X.-S. Yang, "Multiobjective firefly algorithm for continuous optimization," *Engineering with Computers*, vol. 29, no. 2, pp. 175–184, 2013.

[5] Z. Yongquan, L. Jiakun, and Z. Guangwei, "Leader glowworm swarm optimization algorithm for solving nonlinear equations systems," *Przeglad Elektrotechniczny*, vol. 88, no. 1, pp. 101–106, 2012.

[6] A. Mucherino and O. Seref, "Monkey search: a novel metaheuristic search for global optimization," in *Proceedings of the Conference on Data Mining, Systems Analysis, and Optimization in Biomedicine*, pp. 162–173, Gainesville, Fla, USA, March 2007.

[7] K. M. Passino, "Biomimicry of bacterial foraging for distributed optimization and control," *IEEE Control Systems Magazine*, vol. 22, no. 3, pp. 52–67, 2002.

[8] A. R. Mehrabian and C. Lucas, "A novel numerical optimization algorithm inspired from weed colonization," *Ecological Informatics*, vol. 1, no. 4, pp. 355–366, 2006.

[9] R. G. Reynolds, "Cultural algorithms: theory and applications," in *New Ideas in Optimization*, D. W. Corne, M. Dorigo, and F. Glover, Eds., pp. 367–378, McGraw-Hill, Maidenhead, UK, 1999.

[10] B. Alatas, "Chaotic harmony search algorithms," *Applied Mathematics and Computation*, vol. 216, no. 9, pp. 2687–2699, 2010.

[11] X. S. Yang, "Flower pollination algorithm for global optimization," in *Unconventional Computation and Natural Computation*, vol. 7445 of *Lecture Notes in Computer Science*, pp. 240–249, 2012.

[12] I. El-Henawy and M. Ismail, "An improved chaotic flower pollination algorithm for solving large integer programming problems," *International Journal of Digital Content Technology and Its Applications*, vol. 8, no. 3, pp. 72–81, 2014.

[13] O. Abdel Raouf, I. El-henawy, and M. Abdel-Baset, "A novel hybrid flower pollination algorithm with chaotic harmony search for solving sudoku puzzles," *International Journal of Modern Education and Computer Science*, vol. 3, pp. 38–44, 2014.

[14] M. Sharawi, E. Emary, I. A. Saroit, and H. El-Mahdy, "Flower pollination optimization algorithm for wireless sensor network lifetime global optimization," *International Journal of Soft Computing and Engineering*, vol. 4, no. 3, pp. 54–59, 2014.

[15] Wikipedia article on pollination, http://en.wikipedia.org/wiki/Pollination.

[16] B. J. Glover, *Understanding Flowers and Flowering: An Integrated Approach*, Oxford University Press, 2007.

[17] X. S. Yang and S. Deb, "Eagle strategy using Lévy walk and firefly algorithms for stochastic optimization," in *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*, J. R. Gonzalez, D. A. Pelta, C. Cruz, G. Terrazas, and N. Krasnogor, Eds., vol. 284 of *Studies in Computational Intelligence*, pp. 101–111, Springer, Berlin, Germany, 2010.

[18] I. Pavlyukevich, "Lévy flights, non-local search and simulated annealing," *Journal of Computational Physics*, vol. 226, no. 2, pp. 1830–1844, 2007.

[19] S. Das, A. Abraham, U. K. Chakraborty, and A. Konar, "Differential evolution using a neighborhood-based mutation operator," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 3, pp. 526–553, 2009.

[20] X.-S. Yang and S. Deb, "Two-stage eagle strategy with differential evolution," *International Journal of Bio-Inspired Computation*, vol. 4, no. 1, pp. 1–5, 2012.

[21] A. Hedar, "Test function," http://www-optima.amp.i.kyoto-u.ac.jp/member/student/hedar/Hedarfiles/TestGOfiles/Page364.htm.

[22] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, vol. 4, pp. 1942–1948, December 1995.

[23] X. Li and M. Yin, "An opposition-based differential evolution algorithm for permutation flow shop scheduling based on diversity measure," *Advances in Engineering Software*, vol. 55, pp. 10–31, 2013.

[24] D. Simon, "Biogeography-based optimization," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 6, pp. 702–713, 2008.

[25] E. Rashedi, H. Nezamabadi-pour, and S. Saryazdi, "GSA: A Gravitational Search Algorithm," *Information Sciences*, vol. 179, no. 13, pp. 2232–2248, 2009.

[26] X. Yang S, "Firefly algorithm, Lévy flights and global optimization," in *Research and Development in Intelligent Systems XXVI*,

M. Bramer, R. Ellis, and M. Petridis, Eds., pp. 209–218, Springer, London, UK, 2010.

[27] X. S. Yang and S. Deb, “Cuckoo search via Levy flights,” in *Proceedings of the World Congress on Nature & Biologically Inspired Computing*, pp. 210–214, IEEE Publication, New York, NY, USA, 2009.

[28] D. Karaboga and B. Basturk, “A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm,” *Journal of Global Optimization*, vol. 39, no. 3, pp. 459–471, 2007.