Parameter Investigation in Brain Storm Optimization

Zhi-hui Zhan, Wei-neng Chen (Corespoding Author), Ying Lin, Yue-jiao Gong, Yuan-long Li, and Jun Zhang
Department of Computer Science, Sun Yat-sen University, Guangzhou, P. R. China, 510275
Key Laboratory of Digital Life, Ministry of Education, P. R. China
Key Laboratory of Software Technology, Education Dept. of Guangdong Province, P. R. China

***Abstract*—Human being is the most intelligent organism in the world and the brainstorming process popularly used by them has been demonstrated to be a significant and promising way to create great ideas for problem solving. Brain storm optimization (BSO) is a new kind of swarm intelligence algorithm inspired by human being creative problem solving process. BSO transplants the brainstorming process in human being into optimization algorithm design and gains successes. BSO generally uses the *grouping*, *replacing*, and *creating* operators to produce ideas as many as possible to approach the problem solution generation by generation. In these operators, BSO involves mainly three control parameters named: (1) *p_replce* to control the replacing operator; (2) *p_one* to control the creating operator to create new ideas between one cluster and two clusters; and (3) *p_center* (*p_one_center* and *p_two_center*) to control using cluster center or random idea to create new idea. In this paper, we make investigations on these parameters to see how they affect the performance of BSO. More importantly, a new BSO variant designed according to the investigation results is proposed and its performance is evaluated.**

Keywords- Brain storm optimization (BSO); brainstorming process; parameter investigation

I. INTRODUCTION

Brain storm optimization (BSO) is a very new kind of swarm intelligence (SI) [1] algorithm proposed first by Shi in 2011 [2][3]. Like other SI algorithms, BSO is a global optimization technique by emulating the cooperative behaviors of organisms in searching food or in solving problems. However, different from early SI algorithms like ant colony optimization (ACO) [4][5], particle swarm optimization (PSO) [6][7], honey bee optimization (HBO), and bacterial forging optimization (BFO) that emulate the collective behaviors of simple insects or animals like ants, birds, bees, and bacteria, BSO is inspired by the social and swarm behaviors of more intelligent organisms, e.g., the human being. It is intuitive to expect that BSO should be superior to other SI algorithms because BSO emulates the most intelligent animal in the world (human being) instead of simple objects such as ants in ACO, birds in PSO, bees in HBO, and bacteria in BFO.

The BSO algorithm proposed by Shi is motivated by the following intelligent behaviors [2]: when human beings face a difficult problem which every single person may has difficulty to solve, group persons will get together to brainstorm. These persons are usually with different background and they come together for a brainstorming process, which help them to interactively collaborate to generate great ideas. This way, the problem can usually be solved with high probability. Shi has successfully designed a BSO by emulating this brainstorming process in human being solving problem and conducted simulation results on typical benchmark functions to validate the usefulness and effectiveness of BSO in solving optimization problems [2]. Later, Zhan *et al*. [8] modified the grouping operator and the creating operator, so as to design a new BSO variant named modified BSO (MBSO) to enhance the algorithm performance. The comparisons with BSO, PSO, and differential evolution (DE) demonstrate that MBSO is effective and efficient for global optimization. Zhou *et al*. [9] also designed a new version of BSO by modifying the creating operator. Xue *et al*. [10] proposed to use the Cauchy random noise in the creating operator. They applied the BSO algorithm to multi-objective optimization problems.

Generally speaking, BSO consists of three operators: grouping, replacing, and creating, to evolve the ideas generations for problem solving, as shown in Fig. 1.

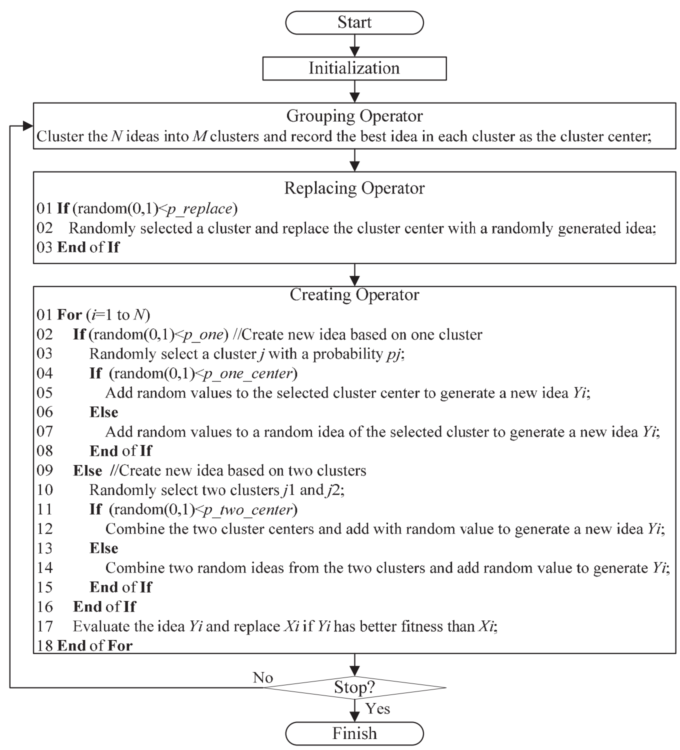


Figure 1. General flowchart of BSO.

This work was supported in part by the National High-Technology Research and Development Program ("863" Program) of China under Grand 2013AA01A212, National Science Fund for Distinguished Young Scholars No. 61125205, National Natural Science Foundation of China No.61070004 and 61202130, NSFC Joint Fund with Guangdong under Key Project U1201258 and U1135005, and the Scholarship Award for Excellent Doctoral Student Granted by Ministry of Education, China.

978-1-4673-6004-3/13/$31.00 ©2013 IEEE

In the initialization operator, BSO randomly generates N ideas ($\boldsymbol{X_i} = [x_{i1}, x_{i2}, \ldots, x_{iD}]$, where $1 \le i \le N$, N is the population size and D is the problem dimension) in the search space as the population. Herein in BSO we refer the individual as idea like naming individual as particle in PSO and calling individual as ant in ACO. In the grouping operator, BSO uses a kind of cluster method, e.g., the *k*-mean cluster method is used in BSO [2] while a simple grouping method (SGM) is adopted in MBSO [8], to group the N ideas into M different clusters. Moreover, the best idea in each cluster is recorded as the cluster center. In the replacing operator, the operation is control by a parameter named $p\_replace \in [0,1]$. A random value in range of [0,1] is generated, if the value is smaller than *p_replace*, then the replacing operator occurs, by randomly selecting a cluster and replacing the cluster center with a randomly generated idea.

After the replacing operation, BSO goes to the creating operator to generate N new ideas ($\boldsymbol{Y_i} = [y_{i1}, y_{i2}, \ldots, y_{iD}]$, $1 \le i \le N$) one by one based on the current ideas. To create a new idea $\boldsymbol{Y_i}$, BSO first determines whether to create the new idea $\boldsymbol{Y_i}$ based on one selected cluster or based on two selected clusters. This is controlled by a parameter named $p\_one \in [0,1]$. If a random generated value is smaller than *p_one*, then the $\boldsymbol{Y_i}$ is created based on one cluster. Otherwise, the $\boldsymbol{Y_i}$ is created based on two clusters. For the first condition, the cluster j is selected according to the number of ideas in each cluster by a roulette selection strategy. That is, the more ideas in the cluster, the larger chance it will be selected. The probability of selecting each cluster j is as:

$$p_j = \frac{|M_j|}{N}, \text{ where } |M_j| \text{ is the number of ideas in cluster } j \quad (1)$$

However, for the second condition, BSO does not select the two clusters according to (1), but just randomly select two clusters $j1$ and $j2$ from the M clusters.

After the cluster(s) have been selected, BSO then uses a parameter $p\_one\_center \in [0,1]$ (when based on one cluster) or a parameter $p\_two\_center \in [0,1]$ (when based on two clusters) to determine whether create the new idea $\boldsymbol{Y_i}$ base on the cluster center(s) or random idea(s) of the cluster(s). No matter to use the cluster center or to use random idea of the cluster, we can regard the selected based idea as $\boldsymbol{X}$, then the new idea $\boldsymbol{Y_i}$ is created as:

$$y_{id} = x_d + \xi_d \times N(\mu,\sigma)_d \quad (2)$$

where d is the dimension index, $N(\mu,\sigma)$ is the Gaussian random value with mean μ and variance σ, and ξ is a coefficient that weights the contribution of the Gaussian random value, which is calculated as:

$$\xi = \text{logsig}((0.5 \times g - G)/k) \times \text{random}(0,1) \quad (3)$$

where logsig() is a logarithmic sigmoid transfer function whose values are within the range (0,1), g and G are the current generation number and maximum number of generation, and k is for changing logsig() function's slope, random(0, 1) is a random value within (0,1).

There are two issues to be reminded. One is that if y_{id} in (2) is out of the search range, its value should be adjusted. In this paper, we just simple set it as the corresponding boundary value which it exceeds. The other issue is that when create the new idea $\boldsymbol{Y_i}$ based on two clusters, the two ideas from the clusters $j1$ and $j2$ first combine themselves and then use Eq. (2) to create $\boldsymbol{Y_i}$. The combination is defined as:

$$\boldsymbol{X} = R \times \boldsymbol{X_1} + (1 - R) \times \boldsymbol{X_2} \quad (4)$$

where R is a random value within (0,1) for all the dimension of $\boldsymbol{X}$, while $\boldsymbol{X_1}$ and $\boldsymbol{X_2}$ are the two ideas from clusters $j1$ and $j2$, respectively.

MBSO modifies the method for creating new idea by using an idea difference strategy (IDS) [8]. When creating a new idea $\boldsymbol{Y_i}$ based on a current idea $\boldsymbol{X}$, two distinct random ideas $\boldsymbol{X_a}$ and $\boldsymbol{X_b}$ from all the current ideas are taken to represent the idea difference, and the $\boldsymbol{Y_i}$ is created as:

$$y_{id} = \begin{cases} random(L_d, H_d), & \text{if } random(0,1) < p_r \\ x_{id} + random(0,1)_d \times (x_{ad} - x_{bd}), & \text{otherwise} \end{cases} \quad (5)$$

where d is the dimension index and $1 \le a \ne b \le N$. The p_r in the IF statement is a new parameter to control the open minded element into the idea creation. The studies in [8] show that p_r=0.005 is a good choice for good algorithm performance.

In the new BSO variant proposed by Zhou *et al*. [9], three new ideas are generated by using one cluster center, two cluster centers, and a random idea reference, respectively.

After the new idea $\boldsymbol{Y_i}$ has been created, BSO evaluates $\boldsymbol{Y_i}$ and replaces $\boldsymbol{X_i}$ if $\boldsymbol{Y_i}$ has a better fitness than $\boldsymbol{X_i}$. This new idea creating process repeats for all the $1 \le i \le N$ to complete a generation. If the termination criteria met, BSO terminates and reports the best idea of the population as the solution. Otherwise, BSO goes to the next generation to repeat the grouping, replacing, and creating processes. It should be noted that in Zhou *et al*. [9]'s BSO, in every generation, each idea creates three different new ideas and three new ideas together with the origin idea compete to survive to the next generation.

From the above descriptions and the Fig. 1, we can see that there are mainly three control parameters in BSO. They are: 1) the *p_replce* to control the replacing operator; 2) the *p_one* to control creating operator between one cluster and two clusters, and 3) the *p_center* (*p_one_center* and *p_two_center*) to control using cluster center or random idea to create new idea.

In this paper, we make investigations on these parameters to see how they affect the performance of BSO and MBSO. As the BSO variant in [9] does not use parameters *p_one* and *p_center*, while the BSO variant in [10] is proposed for multi-objective optimization, they are not investigated in this paper. More important, a new BSO variant is designed according to the investigation results and its performance is evaluated.

The rest of this paper is organized as follows. In Section II, the parameters investigations are conducted on *p_replce*, *p_one*, and *p_center* based on both the BSO and MBSO algorithm. Then Section III proposes a novel simplified MBSO (SMBSO) according to the parameters investigation results of MBSO. Experimental studies for performance evaluation of SMBSO and comparisons are also conducted in Section III. Finally, conclusions are summarized in Section IV. Moreover, some future work directions are also discussed in Section IV.

II. PARAMETERS INVESTIGATIONS

A. Experimental Environments

In order to investigate the influences of different parameters on the algorithm performance, a set of benchmark functions proposed by Yao *et al*. [11] are chosen, as listed in Table I. These 13 functions have been widely used in the evolutionary computation community for algorithm investigations, performance analysis, and comparisons [12][13]. The first 6 functions are unimodal functions and the last 7 are multimodal functions. Moreover, the f_3, f_7, and f_{11} are with strong variables linkage. All the functions are with dimensions D=30.

In all the following investigations, the BSO and MBSO algorithms are implemented by Visual C++ 6.0 and the experiments are run on same machine with Intel Dual 2.40 GHz CPU, 2 GB memory and the Windows 7 operating system. All the parameters are set as the proposals in [2] and [8] (of course not include the parameter which is being investigated). For each function, BSO and MBSO variants with different investigated parameter values (e.g, the *p_replace* values) use the same maximum number of function evaluations (FEs) 3×10^5 in each run. For the purpose of reducing statistical errors, each algorithm is tested 30 times independently for every function and the mean results are used in the comparison. Moreover, when investigating each parameter (e.g., the *p_replace*), we make the Wilcoxon's rank sum tests between the results obtained by BSOs using the proposed value in [2] and [8] (e.g., *p_replace*=0.2 in [2] and [8]) and BSOs using other values, with the significant level of α=0.05 to see that whether the results are statistically different.

B. Investigations on the Replacing Operator

We first investigate the replacing operator. According to Shi's intuition, the replacing operator is to make the population to cover unexplored areas [2]. This operator is controlled by a parameter *p_replace*, which was proposed to be 0.2 in Shi's study [2]. Herein we will set the parameter *p_replace* to be 0, 0.1, 0.2,…, 0.8, 0.9, and 1.0 for both the BSO and MBSO algorithms. The investigation results on the parameter *p_replace* are presented in Table II and Table III for the BSO and MBSO algorithms respectively.

It is very interesting to observe from the tables that the parameter *p_replace* has slight influence on BSO but may affect the performance of MBSO on some functions. Table II shows that the mean solutions obtained by BSO with different *p_replace* values are with the same order of magnitude on most of the functions. For example, no matter the *p_replace* value is 0 or 0.3, or 0.8, or 1.0, or some other values, the mean solutions

TABLE I. BENCHMARK FUNCTIONS

	Benchmark function	Search Space	Global Opt.	f_{min}	Name
Unimodal	$f_1(x)=\sum_{i=1}^{D}x_i^2$	$[-100, 100]^D$	$\{0\}^D$	0	Sphere
	$f_2(x)=\sum_{i=1}^{D}\lvert x_i\rvert+\prod_{i=1}^{D}\lvert x_i\rvert$	$[-10, 10]^D$	$\{0\}^D$	0	Schwefel's 2.22
	$f_3(x)=\sum_{i=1}^{D}(\sum_{j=1}^{i}x_j)^2$	$[-100, 100]^D$	$\{0\}^D$	0	Quadric
	$f_4(x)=\max_i(\lvert x_i\rvert, 1\le i\le D)$	$[-100, 100]^D$	$\{0\}^D$	0	Schwefel's 2.21
	$f_5(x)=\sum_{i=1}^{D}(\lfloor x_i+0.5\rfloor)^2$	$[-100, 100]^D$	$\{0\}^D$	0	Step
	$f_6(x)=\sum_{i=1}^{D}ix_i^4+random[0,1)$	$[-1.28, 1.28]^D$	$\{0\}^D$	0	Noise
Multimodal	$f_7(x)=\sum_{i=1}^{D-1}[100(x_{i+1}-x_i^2)^2+(x_i-1)^2]$	$[-10, 10]^D$	$\{1\}^D$	0	Rosenbrock
	$f_8(x)=\sum_{i=1}^{D}-x_i\sin(\sqrt{\lvert x_i\rvert})+418.9829\times D$	$[-500, 500]^D$	$\{420.9687\}^D$	0	Schwefel
	$f_9(x)=\sum_{i=1}^{D}[x_i^2-10\cos(2\pi x_i)+10]$	$[-5.12, 5.12]^D$	$\{0\}^D$	0	Rastrigin
	$f_{10}(x)=-20\exp(-0.2\sqrt{1/D\sum_{i=1}^{D}x_i^2})-\exp(1/D\sum_{i=1}^{D}\cos 2\pi x_i)+20+e$	$[-32, 32]^D$	$\{0\}^D$	0	Ackley
	$f_{11}(x)=1/4000\sum_{i=1}^{D}x_i^2-\prod_{i=1}^{D}\cos(x_i/\sqrt{i})+1$	$[-600, 600]^D$	$\{0\}^D$	0	Griewank
	$f_{12}(x)=\frac{\pi}{D}\{10\sin^2(\pi y_1)+\sum_{i=1}^{D-1}(y_i-1)^2[1+10\sin^2(\pi y_{i+1})]+(y_D-1)^2\}+\sum_{i=1}^{D}u(x_i,10,100,4)$, where $y_i=1+\frac{1}{4}(x_i+1)$, $u(x_i,a,k,m)=\begin{cases}k(x_i-a)^m, & x_i>a\\ 0, & -a\le x_i\le a\\ k(-x_i-a)^m, & x_i<-a\end{cases}$	$[-50, 50]^D$	$\{1\}^D$	0	Generalized Penalized
	$f_{13}(x)=0.1\{\sin^2(3\pi x_1)+\sum_{i=1}^{D-1}(x_i-1)^2[1+\sin^2(3\pi x_{i+1})]+(x_D-1)[1+\sin^2(2\pi x_D)]\}+\sum_{i=1}^{D}u(x_i,5,100,4)$	$[-50, 50]^D$	$\{1\}^D$	0	Generalized Penalized

TABLE II. MEAN SOLUTIONS COMPARISON ON BSO WITH DIFFERENT *P_REPLACE* VALUES

Func	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
f_1	1.31E-64	1.48E-64	1.41E-64	1.46E-64	1.52E-64	1.62E-64†	1.68E-64†	1.63E-64†	1.65E-64†	1.76E-64†	1.84E-64†
f_2	2.15E-04	3.94E-05	1.68E-03	4.83E-03	1.74E-03	8.01E-04	3.34E-03†	3.70E-03†	4.66E-03†	5.53E-03†	6.60E-03†
f_3	4.87E-01†	3.19E-01	3.62E-01	4.20E-01	3.79E-01	4.61E-01	5.33E-01†	5.75E-01†	5.73E-01†	6.39E-01†	6.99E-01†
f_4	7.34E-03	9.17E-03	9.58E-03	9.15E-03	1.12E-02	1.56E-02†	1.52E-02	2.23E-02†	1.95E-02†	2.35E-02†	2.27E-02†
f_5	0.00E+00	0.00E+00	0.00E+00	0.00E+00	3.33E-02	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_6	1.82E-02†	2.25E-02	2.20E-02	2.11E-02	2.25E-02	2.27E-02	2.59E-02	3.40E-02†	3.51E-02†	3.45E-02†	3.27E-02†
f_7	2.78E+01	3.00E+01	2.81E+01	2.80E+01	2.77E+01	2.78E+01	2.82E+01	2.82E+01	2.82E+01	2.80E+01	2.75E+01
f_8	5.31E+03	5.29E+03	5.35E+03	5.60E+03	5.28E+03	5.33E+03	5.28E+03	5.40E+03	5.40E+03	5.61E+03	5.35E+03
f_9	3.65E+01	2.92E+01†	3.53E+01	3.10E+01†	3.05E+01†	3.27E+01	2.99E+01†	2.98E+01†	3.13E+01†	3.24E+01	2.86E+01†
f_{10}	7.58E-15	7.81E-15	7.46E-15	8.05E-15	8.29E-15	7.81E-15	8.52E-15	8.76E-15†	8.88E-15†	9.23E-15†	8.64E-15†
f_{11}	7.64E-03	1.24E-02	9.77E-03	6.24E-03	1.02E-02	1.41E-02	1.20E-02	6.90E-03	1.07E-02	7.88E-03	1.01E-02
f_{12}	3.48E+00†	2.21E+00	1.01E+00	6.88E-01	3.42E-01	4.40E-01	2.36E-01	5.59E-02†	5.19E-02†	8.24E-02†	9.72E-02†
f_{13}	7.32E-04†	3.66E-04	1.07E-03	1.10E-03†	3.66E-04†	1.43E-03†	2.20E-03†	7.33E-04†	2.08E-07†	3.66E-04†	1.10E-03†

†The results are significantly different by the Wilcoxon's rank sum tests with the significant level of α=0.05

TABLE III. MEAN SOLUTIONS COMPARISON ON MBSO WITH DIFFERENT *P_REPLACE* VALUES

Func	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
f_1	1.69E-94†	3.77E-94†	2.58E-90	7.90E-86†	2.59E-87†	1.02E-84†	9.69E-82†	6.43E-80†	1.88E-77†	5.98E-74†	2.17E-72†
f_2	1.30E-52†	3.77E-52	1.34E-51	7.08E-50†	2.54E-48†	2.25E-48†	1.03E-46†	1.29E-45†	1.19E-44†	1.56E-43†	1.08E-42†
f_3	6.53E-25†	1.40E-24†	5.12E-23	1.04E-21†	1.15E-21†	1.27E-20†	2.74E-20†	6.69E-19†	2.26E-18†	6.81E-17†	3.32E-16†
f_4	1.24E-01†	8.78E-02	8.69E-02	8.07E-02†	4.77E-02†	3.73E-02†	3.37E-02†	3.44E-02†	2.31E-02†	1.76E-02†	1.70E-02†
f_5	2.00E-01	2.33E-01	3.33E-01	2.00E-01	1.33E-01	5.00E-01	2.00E-01	2.00E-01	2.33E-01	2.00E-01	3.00E-01
f_6	5.29E-03†	5.92E-03†	9.13E-03	1.33E-02†	1.62E-02†	2.06E-02†	2.68E-02†	3.01E-02†	3.42E-02†	3.61E-02†	4.31E-02†
f_7	1.69E-01	1.43E-01	2.63E-01	4.33E-01	6.84E-01	1.13E+00	9.57E-01	1.66E+00†	2.14E+00†	3.18E+00†	2.59E+00†
f_8	3.82E-04	3.82E-04	3.82E-04	3.82E-04	3.82E-04	3.82E-04	3.82E-04	3.82E-04	3.82E-04	3.82E-04	3.82E-04
f_9	1.48E-15	6.57E-15	1.18E-15	5.33E-16	6.51E-16	1.48E-15	5.33E-16	7.51E+00	6.53E+00	6.89E+01	5.39E+01†
f_{10}	1.17E-14†	9.83E-15	8.64E-15	8.64E-15	8.05E-15	7.58E-15	7.58E-15	6.87E-15†	7.69E-15	7.22E-15†	7.34E-15†
f_{11}	1.75E-02	1.84E-02	1.68E-02	1.96E-02	2.03E-02	1.61E-02	1.11E-02	1.98E-02	1.80E-02	1.00E-02	2.07E-02
f_{12}	2.72E-32	7.25E-32	2.21E-32	2.10E-32	1.78E-32	2.21E-32	2.76E-32†	1.64E-32†	2.26E-32	1.88E-32	6.06E-32
f_{13}	1.46E-03	2.56E-03	2.93E-03	2.56E-03	1.83E-03	2.20E-03	7.32E-04†	2.20E-03	2.56E-03	1.83E-03	3.30E-03

†The results are significantly different by the Wilcoxon's rank sum tests with the significant level of α=0.05

to f_1 are with E-64 order of magnitude, and the mean solutions to f_9 are with E+01 order of magnitude. However, for MBSO, as shown in Table III, the mean solutions obtained under different *p_replace* values are much different. For example, the solution accuracy for f_1 becomes worse as the increase of the *p_replace* value. Such phenomenon is also observed on other unimodal functions like f_2, f_3, and f_6. This is not much surprising because the purpose of the replacing operator is for bringing random noise into the algorithm to avoid local optimal. Therefore, it is not surprising that the replacing operator makes no or small contribution on unimodal functions. We also plot the investigation results of *p_replace* of BSO on the unimodal functions f_1, f_2, and f_3 in Fig. 2 to make the results clearer

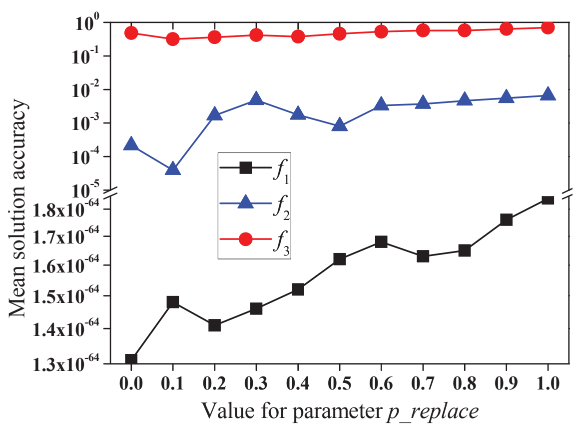
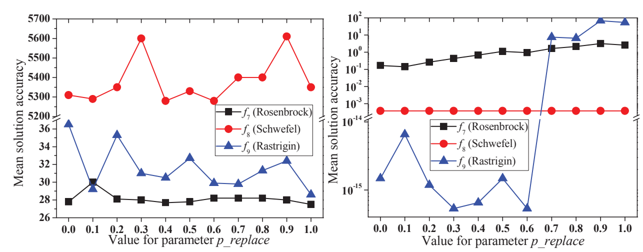


Figure 2. Parameter investigation results of *p_replace* of BSO on some unimodal functions.



(a) BSO (b) MBSO

Figure 3. Parameter investigation results of *p_replace* of (a) BSO and (b) MBSO on some multimodal functions.

When for multimodal functions, an interesting observation is that the replacing operator seems cannot make BSO out of local optima, e.g., the results on the Rosenbrock (f_7), Schwefel (f_8), and Rastrigin (f_9) functions are not much different, as shown in Table II. However, results on f_{12} and f_{13} show that larger *p_replace* value seems to be better sometimes. When for MBSO, the results in Table III even show that the MBSO performance becomes worse as the *p_replace* value increases. This may be due to that MBSO itself has the ability to avoid local optimal of the multimodal functions. The replacing operator proposed in [2] just replaces a randomly selected cluster center by a randomly generated idea. However, this may be harmful in some conditions that a good idea is replaced by a bad idea. Therefore, the results in Tables II&III and the curves in Fig. 3 for multimodal functions f_7, f_8, and f_9 indicate that without or less using the replacing operator (with not very large *p_replace* value), BSO and MBSO can also do as well as, or even better than BSO and MBSO with replacing operator.

TABLE IV. MEAN SOLUTIONS COMPARISON ON BSO WITH DIFFERENT *P_ONE* VALUES

Func	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
f_1	7.13E-65†	7.80E-65†	8.37E-65†	8.33E-65†	9.11E-65†	9.58E-65†	1.09E-64†	1.26E-64†	1.41E-64	1.75E-64†	2.40E-64†
f_2	3.79E-32†	3.92E-32†	4.01E-32†	4.12E-32†	4.26E-32†	1.65E-22†	9.54E-06†	9.20E-05†	1.37E-03	2.50E-03†	6.79E-02†
f_3	2.20E-01†	2.15E-01†	2.26E-01†	2.23E-01†	2.21E-01†	2.37E-01†	2.52E-01†	2.53E-01†	3.32E-01	4.90E-01†	9.69E-01†
f_4	6.01E-07†	8.90E-06†	7.36E-06†	8.55E-05†	3.58E-04†	3.49E-04†	2.17E-03†	2.84E-03†	9.29E-03	2.41E-02†	7.96E-02†
f_5	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	3.00E-01†
f_6	3.82E-03†	4.04E-03†	4.18E-03†	6.07E-03†	6.29E-03†	8.00E-03†	1.04E-02†	1.43E-02†	2.26E-02	3.99E-02†	1.04E-01†
f_7	2.80E+01	2.80E+01	2.80E+01	2.81E+01	2.81E+01	2.80E+01	2.82E+01	2.79E+01	2.82E+01	3.48E+01	3.66E+01
f_8	5.59E+03	5.25E+03	5.45E+03	5.30E+03	5.29E+03	5.39E+03	5.28E+03	5.34E+03	5.41E+03	5.41E+03	5.39E+03
f_9	2.24E+01†	2.36E+01†	2.37E+01†	2.53E+01†	2.51E+01†	3.01E+01	2.83E+01†	3.09E+01	3.28E+01	3.09E+01	3.62E+01†
f_{10}	4.14E-15†	4.14E-15†	4.14E-15†	4.26E-15†	4.14E-15†	4.97E-15†	5.56E-15†	6.63E-15	7.69E-15	9.94E-15†	7.78E-02†
f_{11}	8.12E-03	7.23E-03	1.05E-02	6.73E-03†	7.38E-03	8.12E-03	1.06E-02	8.87E-03	1.08E-02	1.00E-02	6.16E-03
f_{12}	8.22E-01	3.29E-01†	5.63E-01	6.33E-01	6.40E-01	1.10E+00	9.77E-01	8.41E-01	1.04E+00	1.76E+00†	3.61E+00†
f_{13}	3.66E-04†	1.35E-32†	1.35E-32†	1.35E-32†	3.66E-04†	1.35E-32†	1.35E-32†	7.32E-04†	3.66E-04	1.46E-03†	1.83E-03†

†The results are significantly different by the Wilcoxon's rank sum tests with the significant level of α=0.05

TABLE V. MEAN SOLUTIONS COMPARISON ON MBSO WITH DIFFERENT *P_ONE* VALUES

Func	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
f_1	6.55E-11†	1.08E-23†	1.88E-41†	2.84E-55†	1.83E-65†	7.30E-74†	6.07E-81†	8.20E-86†	3.35E-92	5.83E-94†	5.45E-98†
f_2	3.92E-05†	1.73E-10†	7.32E-18†	6.66E-28†	4.35E-35†	1.79E-40†	2.31E-42†	1.71E-47†	4.58E-51	4.39E-54†	5.73E-55†
f_3	6.91E-02†	3.37E-06†	1.02E-09†	6.66E-13†	8.32E-16†	2.47E-17†	2.39E-20†	6.05E-21†	1.92E-23	6.69E-25†	4.10E-26†
f_4	3.50E+00†	3.45E+00†	2.80E+00†	1.95E+00†	1.15E+00†	6.54E-01†	3.23E-01†	1.85E-01†	6.98E-02	5.89E-02	8.20E-02
f_5	2.80E+00†	9.00E-01†	6.00E-01	6.33E-01	4.00E-01	3.67E-01	1.33E-01†	3.67E-01	4.33E-01	3.00E-01	3.00E-01
f_6	2.69E-02†	1.89E-02†	1.59E-02†	1.34E-02†	1.16E-02†	1.13E-02†	9.68E-03	1.05E-02	9.56E-03	8.93E-03	8.77E-03
f_7	3.14E+01†	2.40E+01†	7.35E+00†	5.10E+00†	2.95E+00†	2.48E+00†	1.64E+00†	5.43E-01	2.34E-01	2.44E-01	1.89E-01
f_8	2.64E+03†	3.82E-04	3.82E-04	3.82E-04	3.82E-04	3.82E-04	3.82E-04	3.82E-04	3.82E-04	3.82E-04	3.82E-04
f_9	1.76E+02†	1.16E+02†	2.88E+01†	6.59E+00†	1.72E-15	2.49E-15	2.01E-15	2.25E-15	1.42E-15	1.01E-15	1.95E-15
f_{10}	3.36E-02†	1.45E-10†	3.34E-14†	1.72E-14†	1.55E-14†	1.11E-14	1.10E-14	9.71E-15	9.83E-15	8.52E-15	9.59E-15
f_{11}	2.09E-02†	1.98E-02†	2.91E-02†	1.43E-02†	1.81E-02†	1.64E-02	1.59E-02	1.94E-02	8.78E-03	1.89E-02	1.60E-02
f_{12}	3.46E-03†	2.52E-15†	1.64E-32	1.87E-32	1.85E-32	2.14E-32	1.97E-32	2.93E-32	2.36E-32	3.12E-32	1.50E-31
f_{13}	5.06E-03†	1.43E-03†*	1.80E-03	2.20E-03	1.83E-03	2.56E-03	2.53E-03	2.20E-03	1.43E-03	2.56E-03	2.93E-03†

†The results are significantly different by the Wilcoxon's rank sum tests with the significant level of α=0.05

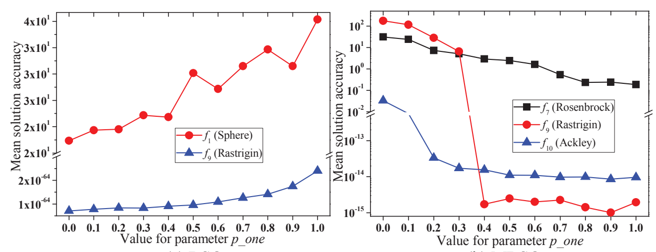
*Although the mean values are the same, the results obtained by *p_one*=0.8 are significantly better than those of *p_one*=0.1

Nevertheless, as a mechanism to introduce diversity to the algorithm, the replacing operator may produce positive effect if it is well design, e.g., maybe we can replace the worst idea of the cluster but not the cluster center (the best idea of the cluster). However, this research topic is out of the scope of this paper and can be further work.

C. Using One Cluster or Two Clusters to Create New Idea?

Another interesting parameter of BSO is the *p_one* which is used to control whether to create the new idea based on one cluster or based on two clusters. In order to investigate how this parameter affects the algorithm performance, we set its value to 0, 0.1,…, 0.8, 0.9, and 1.0, with other parameters the same as those in [2] and [8]. The experimental results are presented in Tables IV&V for BSO and MBSO respectively. As the *p_one* value was set to 0.8 in [2] and [8], we make the Wilcoxon's rank sum tests between the results obtained by *p_one*=0.8 and *p_one* with other values, with the significant level of α=0.05 to see that whether the results are statistically different.

This parameter exhibits different influences on BSO and MBSO. For BSO, a relative small *p_one* value seems to be better while a relative large *p_one* value is better for MBSO, as shown in Fig. 4(a). The reasons may be that when the *p_one* value is smaller, BSO has larger chance to create new ideas based on two clusters, which is helpful for the algorithm to utilize more information from the population to obtain better performance, especially on f_2, f_4, f_6, f_9, f_{10}, f_{12}, and f_{13}.



(a) BSO (b) MBSO

Figure 4. Parameter investigation results of *p_one* of (a) BSO and (b) MBSO on some functions.

However, for MBSO, the IDS in the creating operator itself has the ability to utilize more information from the population to match the search environment to provide suitable disturbed values for creating better ideas [8]. Therefore, using more clusters (small *p_one* values) to create new ideas may slow down the convergence speed. The results in Table V and Fig. 4(b) clearly show that large *p_one* values not only make MBSO performs significantly better than MBSO with small *p_one* values on unimodal functions, but also on multimodal functions. This makes us rationally to think that setting *p_one*=1.0, i.e., using only one cluster to create new idea, is a promising choice for MBSO.

Nevertheless, we still remind that as a parameter to control exploration and exploitation, the influence of *p_one* values is needed more and further investigations under different environments.

TABLE VI. MEAN SOLUTIONS COMPARISON ON BSO WITH DIFFERENT *P_ONE_CENTER* VALUES

Func	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
f_1	1.23E-09†	1.88E-64†	1.64E-64†	1.45E-64†	1.25E-64	1.26E-64	1.23E-64	1.06E-64†	1.08E-64†	1.05E-64†	1.03E-64†
f_2	1.54E-02†	8.31E-03†	3.44E-04†	3.39E-03†	8.10E-04	8.22E-04†	3.18E-05†	3.10E-04†	1.44E-04†	5.07E-06†	6.96E-06†
f_3	8.49E-01†	6.87E-01†	5.47E-01†	4.95E-01	4.00E-01	4.22E-01	3.72E-01	3.48E-01	2.95E-01†	4.06E-01	3.51E-01
f_4	1.88E-02†	1.26E-02	1.02E-02	8.39E-03	9.10E-03	7.71E-03	5.89E-03†	3.93E-03†	4.21E-03†	6.01E-03†	4.01E-03†
f_5	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_6	1.14E-02†	1.44E-02	1.36E-02	1.92E-02	1.66E-02	2.02E-02	2.07E-02	2.13E-02	2.11E-02†	2.61E-02†	2.65E-02†
f_7	2.81E+01	2.82E+01	2.80E+01	2.81E+01	2.82E+01	2.75E+01†	2.78E+01	2.77E+01†	2.99E+01	2.74E+01†	3.47E+01
f_8	5.58E+03	5.31E+03	5.41E+03	5.44E+03	5.36E+03	5.31E+03	5.35E+03	5.36E+03	5.41E+03	5.39E+03	5.30E+03
f_9	2.89E+01†	3.11E+01†	3.43E+01	3.36E+01	3.66E+01	3.35E+01	4.06E+01†	3.93E+01	3.87E+01	3.99E+01	4.53E+01†
f_{10}	8.31E-06†	8.76E-15†	8.17E-15	7.69E-15	7.69E-15	7.58E-15	7.34E-15	7.34E-15	7.34E-15	7.81E-15	7.34E-15
f_{11}	1.04E-02	8.21E-03	7.15E-03	1.11E-02	8.53E-03	1.21E-02	9.19E-03	1.13E-02	7.55E-03	8.29E-03	6.57E-03†
f_{12}	5.40E-01†	1.18E+00†	2.23E+00	2.67E+00	3.47E+00	3.63E+00	4.86E+00†	5.83E+00†	5.18E+00†	5.09E+00†	6.19E+00†
f_{13}	4.01E-04†	1.84E-06†	7.33E-04†	3.66E-04†	1.35E-32	7.32E-04	3.66E-04	3.66E-04	1.35E-32	1.10E-03	3.66E-04

†The results are significantly different by the Wilcoxon's rank sum tests with the significant level of α=0.05

TABLE VII. MEAN SOLUTIONS COMPARISON ON MBSO WITH DIFFERENT *P_ONE_CENTER* VALUES

Func	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
f_1	2.32E-33†	3.04E-53†	9.42E-70†	2.55E-83†	1.69E-90	6.94E-89†	3.89E-84†	3.49E-57†	1.35E-30†	1.49E-13†	4.53E-09†
f_2	2.52E-17†	1.04E-28†	2.98E-39†	5.07E-48†	2.14E-50	5.35E-49†	2.87E-36†	2.41E-26†	4.62E-15†	1.74E-08†	3.73E-08†
f_3	1.52E-01†	1.06E-07†	2.68E-14†	1.45E-19†	7.04E-23	8.92E-26†	9.18E-24†	1.82E-15†	3.98E-03†	1.11E+00†	1.31E+01†
f_4	2.08E-03†	8.31E-05†	9.35E-04†	7.02E-03†	9.87E-02	5.09E-01†	1.65E+00†	3.29E+00†	4.78E+00†	6.88E+00†	9.53E+00†
f_5	0.00E+00†	1.00E-01	1.33E-01	3.33E-01	1.67E-01	2.33E-01	2.33E-01	1.00E-01	3.00E-01	1.67E-01	1.67E-01
f_6	1.61E-02†	1.52E-02†	1.21E-02	1.09E-02	1.02E-02	9.04E-03	7.68E-03†	7.56E-03†	7.05E-03†	6.58E-03†	6.71E-03†
f_7	1.07E+01†	4.04E+00†	2.31E+00†	9.29E-01†	2.50E-01	1.97E-01	5.85E-01	1.88E+00	9.51E+00†	2.69E+01†	3.91E+01†
f_8	2.13E+01†	3.82E-04	3.82E-04	3.82E-04	3.82E-04	3.82E-04	3.82E-04	3.82E-04	3.82E-04†	1.45E-03†	5.31E-03†
f_9	4.18E+01†	5.17E-06	0.00E+00†	2.96E-16	1.78E-15	7.82E-15†	1.53E-14†	1.34E-11†	1.21E-07†	6.87E-05†	1.26E-03†
f_{10}	6.39E-15†	5.33E-15†	6.04E-15†	6.75E-15†	8.88E-15	1.69E-14†	2.65E-14†	1.37E-11†	1.92E-06†	1.75E-03†	4.72E-03†
f_{11}	2.47E-04†	4.43E-03†	1.19E-02	1.45E-02	1.28E-02	1.81E-02	2.89E-02†	3.94E-02†	3.43E-02†	5.09E-02†	7.04E-02†
f_{12}	1.57E-32†	1.57E-32†	6.92E-32	1.79E-32	2.02E-32	9.11E-32†	1.95E-31†	2.46E-31†	3.87E-17†	7.23E-14†	7.33E-10†
f_{13}	3.66E-04	1.46E-03	1.46E-03	2.56E-03	2.56E-03	1.46E-03	1.46E-03	2.93E-03	3.30E-03	1.46E-03†	1.83E-03†

†The results are significantly different by the Wilcoxon's rank sum tests with the significant level of α=0.05

D. Using Cluster Center or Random Idea to Create New Idea?

The last parameter investigated in this paper is the *p_center* that control whether to use cluster center or random idea to create new idea. Notice that there are two kinds of *p_center*, i.e., the *p_one_center* and the *p_two_center* in BSO. Herein the results presented in Table VI and Table VII are based on *p_one_center* for BSO and MBSO, respectively.

For the BSO algorithm, it can be observed that larger *p_one_center* values can generally perform better than smaller *p_one_center* values on unimodal functions. However, when solving multimodal functions, it seems that relatively smaller *p_one_center* values may be better. This may be because that the *p_one_center* value can change the exploration and exploitation abilities. When *p_one_center* value is large, BSO is more likely to use cluster center to create new ideas, this is helpful to accelerate the convergence speed and result in good performance on unimodal functions. When *p_one_center* value is small, BSO can use more information of the population to create new ideas and therefore is good for multimodal functions. However, *p_one_center*=0 is harmful to the algorithm performance.

An interesting observation from the Table VI and Table VII is that the *p_one_center* value has evident influences on MBSO than on BSO. Although the results obtained by BSO with different *p_one_center* values may be significantly different, most of the mean solutions are in the same order of magnitude. However, for the MBSO algorithm, the mean solutions obtained by different *p_one_center* values not only are significantly different by the Wilcoxon's rank sum test, but also are with evidently different orders of magnitudes as shown in Table VII.

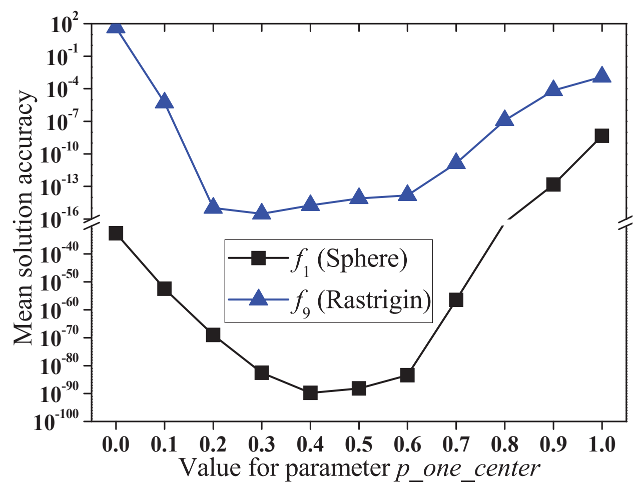


Figure 5. Parameter investigation results of *p_one_center* of MBSO on unimodal function f_1 and multimodal function f_9.

Another very interesting observation from Table VII is that: when using MBSO, it is not small or large *p_one_center* value has good performance, but *p_one_center* with somehow median value is good at most of the functions. For example, as shown in Fig. 5, the solution accuracy of f_1 gets better as the *p_one_center* value increasing from 0 to 0.4, but then gets worse as the parameter increasing from 0.4 to 1.0. The investigation results also show that *p_one_center*=0.4, which is adopted in [2] and [8], is very promising on most of the functions. The curve of the multimodal function f_1 also shows that *p_one_center* with a median value has good performance. This may be due to that a median *p_one_center* value can balance the exploration and exploitation abilities. However, as this parameter is sensitive to the performance, how to efficiently control its value to enhance the algorithm is still a significant future work in BSO.

We also conduct investigations on the *p_two_center* values and find that this parameter affects the BSO and MBSO performance slightly. Therefore we do not prsesent the results herein.

III. SIMPIFIED MBSO

A. Algorithm Design

According to the parameters investigations in the above section, we can summary that:

1) The current replacing operator makes less or even no contributions to the BSO and MBSO algorithms, even though more elaborately designed replacing operator may work better;

2) Using more clusters to create new ideas can sometimes be better than using only one cluster to create new ideas in BSO. However, in MBSO, it seems that using only one cluster to create new ideas is efficient enough and can obtain better performance;

3) The *p_center* parameter can affect the exploration and exploitation abilities. Large *p_center* value is good for unimodal functions because this can make the algorithm converge fast, while small *p_center* value is good for multimodal functions because this can make the algorithm not easy to be trapped by the local optima of the cluster centers.

Based on the above summaries, we design a simplified MBSO (SMBSO) herein to make the algorithm easier for implementation and at the same time is expected to enhance the algorithm performance. We call the new BSO variant SMBSO because it is based on the MBSO algorithm and at the same time makes the algorithm simper and easier. The first simplification is to remove the replacing operator from MBSO. This is rational because that the replacing operator seems to contribute less to the MBSO performance, as observing from our investigations that *p_replace*=0 performs generally better. The second simplification is to use only one cluster in the creating operator to generate new ideas. This is because that our investigations have shown that *p_one*=1.0 seems to be better than others. The third modification of SMBSO is that when every time determine whether to use the cluster center or random idea to create new idea, the parameter *p_center* is set to a random value according to a normal (Gaussian) distribution with mean 0.4 and standard deviation 0.1

$$p\_center = N(0.4,0.1) \tag{6}$$

where the value of 0.4 is adopted directly from [2] and [8], and also our above investigations. This modification is expected to make the algorithm samples more parameter values around the promising region. One thing should be noticed is that herein the *p_center* is referred to *p_one_center* because SMSBO only uses one cluster to create new idea (as *p_one*=1.0).

B. Performance Evalutions

In order to evaluate the SMBSO performance, we test it on all the 13 functions of Table I. Moreover, we not only compare the SMBSO algorithm with the BSO [2] in and the MBSO in [8], but also the current popular PSO and DE algorithms. The maximal FEs for each function is 3×10^5 and the run times is 30.

The solutions obtained by each algorithm are compared in Table VIII. The better results are marked with **boldface**. The Wilcoxon's rank sum tests with the significant level of α=0.05 between SMBSO and other algorithms are also given.

The results show that SMBSO does better than MBSO on all the functions. For the unimodal functions, SMBSO obtains the best results among all the five algorithms on f_1, f_2, f_3, and f_5. For the multimodal functions, Table 9 also shows that SMBSO is the best algorithm for f_5, f_6, and f_7. More important, it can be observed from the table that only MBSO and SMBSO can obtain the global optima of the difficult multimodal functions as the Rosenbrock's function (f_7), the Schwefel's function (f_8), and the Rastrigin's function (f_9).

TABLE VIII. SOLUTIONS ACCURACY (MEAN AND STANDARD DEVIATION) COMPARISONS

Function	PSO: ω:0.9~0.4, c_1= c_2=2, V_{MAXd}=0.2×range, N=40, global version	DE: F=0.5, CR=0.5, N=50, rand/1 mutation scheme	BSO: k=20, μ=0, σ=1	MBSO: p_r=0.005	SMBSO: N=100, M=5, *p_one_center*= N(0.4,0.1)
			p_replace=0.2, *p_one*=0.8, N=100, M=5, *p_one_center*=0.4, *p_two_center*=0.5		
f_1	5.36E-48±1.64E-47†	1.29E-87±1.78E-87†	1.50E-64±3.02E-65†	6.13E-91±2.62E-90†	**3.96E-103±1.00E-102**
f_2	1.77E-32±3.62E-32†	2.83E-50±2.71E-50†	9.93E-04±3.00E-03†	2.32E-51±9.80E-51†	**1.22E-57±2.75E-57**
f_3	1.09E-01±9.31E-02†	4.75E+03±1.22E+03†	3.73E-01±1.60E-01†	3.00E-23±6.87E-23†	**1.67E-27±6.85E-27**
f_4	1.87E-01±1.25E-01†	**5.65E-12±1.45E-12†**	7.35E-03±6.89E-03†	7.78E-02±6.60E-02	6.30E-02±4.13E-02
f_5	**0.00E+00±0.00E+00**	**0.00E+00±0.00E+00**	**0.00E+00±0.00E+00**	1.67E-01±4.61E-01†	**0.00E+00±0.00E+00**
f_6	6.15E-03±2.25E-03†	**3.45E-03±8.34E-04†**	1.95E-02±7.41E-03†	9.27E-03±2.58E-03†	4.19E-03±1.07E-03
f_7	3.27E+01±2.60E+01†	1.73E+01±1.08E+00†	2.79E+01±7.68E-01†	1.46E-01±3.86E-01	**3.24E-02±6.52E-02**
f_8	2.37E+03±3.98E+02†	1.78E+02±1.38E+02†	5.41E+03±7.03E+02†	**3.82E-04±1.30E-12**	**3.82E-04±1.41E-12**
f_9	2.56E+01±7.20E+00†	9.28E+01±7.52E+00†	3.08E+01±7.93E+00†	1.36E-15±2.27E-15	**1.07E-15±1.66E-15**
f_{10}	1.09E-14±2.35E-15	**4.14E-15±0.00E+00†**	7.58E-15±1.47E-15†	1.33E-14±2.46E-14†	1.27E-14±4.23E-15
f_{11}	1.22E-02±1.50E-02	**0.00E+00±0.00E+00†**	8.61E-03±9.97E-03	2.31E-02±2.85E-02	1.30E-02±1.56E-02
f_{12}	3.46E-03±1.89E-02†	**1.57E-32±2.78E-48†**	1.48E+00±1.58E+00†	3.05E-32±5.92E-32†	5.07E-32±8.59E-32
f_{13}	7.32E-04±2.79E-03	**1.35E-32±5.57E-48†**	3.66E-04±2.01E-03	1.83E-03±4.16E-03†	1.83E-03±4.16E-03

†T he results are significantly different with those obtained by SMBSO by the Wilcoxon's rank sum tests with the significant level of α=0.05

IV. Conclusions

In this paper, we have analyzed the impacts of three major parameters in BSO, i.e., the *p_replce*, *p_one*, and *p_center*, based on both the BSO and MBSO algorithm over a set of benchmark functions.

Experimental results conclude that the *p_replce* value has slight influence on the algorithms performance. Therefore:

1) Designing BSO variant without using replacing operator may somehow does not degenerate the performance but makes the algorithm simpler. Even more, better performance may sometimes be expected.

2) Designing a more efficient replacing operator is a significant and challenge future work in BSO.

The results also show that the *p_one* value has evident influences on BSO rather than MBSO. A relative large *p_one* value is good for MBSO.

At last, the investigations on *p_center* show that this parameter affects the MBSO performance more evidently than BSO. A median value near 0.4 would be good for balancing the exploration and exploitation abilities.

More importantly, we have designed a simplified MBSO according to the investigations results. Even though good experimental results are obtained by SMBSO, the search behaviors of BSO and parameters selection for BSO are still needed further studied. By doing so, a clearer understanding of BSO performance will be obtained. Adaptive parameter control strategies [14][15], self-adaptive parameter control strategies [16], and machine learning techniques [17] for online tuning parameters are worthy future research topics. Moreover, further comparisons with other new algorithm variants like the differential ant-stigmergy algorithm [18] are also needed to be studied.

References

[1] J. Kennedy, R. C. Eberhart, and Y. H. Shi, *Swarm Intelligence*, San Mateo, CA: Morgan Kaufmann, 2001.

[2] Y. Shi, "Brain storm optimization algorithm," in *Proc. 2nd Int. Conf. on Swarm Intelligence*, 2011, pp. 303-309.

[3] Y. Shi, "An optimizaiton algorithm based on brainstorming process," *International Journal of Swarm Intelligence Research*, vol. 2, no. 4, pp. 35-62, Oct-Dec. 2011.

[4] W. N. Chen and J. Zhang, "Ant colony optimization approach to grid workflow scheduling problem with various QoS requirements," *IEEE Trans. Syst., Man, and Cybern. C.*, vol. 39, no. 1, pp. 29-43, Jan. 2009.

[5] Z. H. Zhan, J. Zhang, Y. Li, O. Liu, S. K. Kwok, W. H. Ip, and O. Kaynak, "An Efficient Ant Colony System Based on Receding Horizon Control for the Aircraft Arrival Sequencing and Scheduling Problem," *IEEE Trans. Intell. Transp. Syst.*, vol. 11, no. 2, pp. 399-412, Jun. 2010.

[6] W. N. Chen, J. Zhang, H. Chung, W. L. Zhong, W. G. Wu, and Y. H. Shi, "A novel set-based particle swarm optimization method for discrete optimization problems," *IEEE Trans. Evol. Comput.*, vol. 14, no. 2, pp. 278-300, April 2010.

[7] Z. H. Zhan, J. Li, J. Cao, J. Zhang, H. Chung, and Y. H. Shi, "Multiple populations for multiple objectives: A coevolutionary technique for solving multiobjective optimization problems," *IEEE Trans. Syst., Man, and Cybern. B*, In press 2012.

[8] Z. H. Zhan, J. Zhang, Y. H. Shi, and H. L. Liu, "A modified brain storm optimization," in *Proc. IEEE Cong. Evol. Comput.* 2012, pp. 1-8.

[9] D. D. Zhou, Y. H. Shi, and S. Cheng, "Brain storm optimization algorithm with modified step-size and individual generation," in *Proc. Int. Conf. Swarm Intelligence*, 2012, pp. 243-252.

[10] J. Q. Xue, Y. Y. Wu, Y. H. Shi, and S. Cheng, "Brain storm optimization algorithm for multi-objective optimization problems," in *Proc. Int. Conf. Swarm Intelligence*, 2012, pp. 513-519.

[11] X. Yao, Y. Liu and G. M. Lin, "Evolutionary programming made faster," *IEEE Trans. Evol. Comput.*, vol. 3, no. 2, pp. 82-102, Jul. 1999.

[12] J. Zhang, H. Chung, and W. L. Lo, "Clustering-based adaptive crossover and mutation probabilities for genetic algorithms," *IEEE Trans. Evol. Comput.*, vol. 11, no. 3, pp. 326-335, Jun. 2007.

[13] Z. H. Zhan, J. Zhang, Y. Li, and Y. H. Shi, "Orthogonal learning particle swarm optimization," *IEEE Trans. Evol. Comput.*, vol. 15, no. 6, pp. 832-847, Dec. 2011.

[14] Z. H. Zhan, J. Zhang, Y. Li, and H. S. H. Chung, "Adaptive particle swarm optimization," *IEEE Trans. Syst., Man, and Cybern. B.*, vol. 39, no. 6, pp. 1362-1381, Dec. 2009.

[15] J. Q. Zhang and A. C. Sanderson, "JADE: Adaptive differential evolution with optional external archive," *IEEE Trans. Evol. Comput.*, vol. 13, no. 5, pp. 945–958, Oct. 2009.

[16] Z. H. Zhan and J. Zhang, "Self-adaptive differential evolution based on PSO learning strategy," in *Proc. Genetic Evol. Comput. Conf.*, Portland, America, Jul., 2010, pp. 39-46.

[17] J. Zhang, Z. H. Zhan, Y. Lin, N. Chen, Y. J. Gong, J. H. Zhong, H. Chung, Y. Li, and Y. H. Shi, "Evolutionary computation meets machine learning: A survey," *IEEE Computational Intelligence Magazine*, vol. 6, no. 4, pp. 68-75, Nov. 2011.

[18] P. Korošec, J. Šilc, and B. Filipič, "The differential ant-stigmergy algorithm," *Information Sciences*, vol. 192, pp. 82-97, 2012.