Determinative Brain Storm Optimization

Georgia Sovatzidi and Dimitris K. Iakovidis(✉)

University of Thessaly, Papasiopoulou Street 2-4, 35131 Lamia, Greece
g.sovatzidi@gmail.com, diakovidis@uth.gr

Abstract. Brain Storm Optimization (BSO) is a swarm intelligence optimization algorithm, based on the human brainstorming process. The ideas of a brainstorming process comprise the solutions of the algorithm, which iteratively applies solution grouping, generation and selection operators. Several modifications of BSO have been proposed to enhance its performance. In this paper, we propose a novel modification enabling faster convergence of BSO to optimal solutions, without requiring setting an upper bound of algorithm iterations. It considers a brainstorming scenario where participating groups with similar ideas recognize that their ideas are similar, and together, collaborate for the determination of a better solution. The proposed modification, called Determinative BSO (DBSO), implements this scenario by applying a cluster merging strategy for merging groups of similar solutions, while following elitist selection. Experimental results using eleven benchmark functions show that the proposed modified BSO performs better than both the original and a state-of-the-art algorithm.

Keywords: Brain Storm Optimization · Swarm intelligence · Cluster merging

1 Introduction

Motivated by the way people cooperate to solve problems Shi [24] proposed a metaheuristic, swarm optimization algorithm, named Brain Storm Optimization (BSO) algorithm. BSO is a recent algorithm inspired by the human brainstorming process, which obeys the Osborn's four rules [20]. The members of a group that participate in the generation of an idea have to be open-minded and with an, as much as possible, diverse background. BSO possesses a great potential as an optimization tool and it has already been applied to solve successfully problems in various domains. Some of these applications include optimal design of efficient motors [8], optimization of satellite formation and reconfiguration [27], optimization of coverage and connectivity of wireless sensor networks [22], image fusion [19], prediction of protein folding kinetics [1], classification [15], and stock index forecasting [30].

Since the introduction of BSO, in 2011, many attempts have been made to improve its performance. Such an attempt is the Modified BSO (MBSO) [36], which uses a simple grouping method (SGM) for grouping ideas, instead of the *k*-means clustering algorithm used in the original BSO. Another attempt is the Quantum-behaved BSO (QBSO) [9], which has been proposed to cope with entrapment in local optima by an approach inspired by quantum theory. Zhou *et al.* [37] introduced an adaptive step-size coefficient,

© Springer Nature Switzerland AG 2020
Y. Tan et al. (Eds.): ICSI 2020, LNCS 12145, pp. 259–271, 2020.
https://doi.org/10.1007/978-3-030-53956-6_24

which can be utilized to balance the convergence speed of the algorithm. Various solution, generation and selection strategies have been proposed, mainly aiming to maintain the diversity for the whole population. In [35] two different mutation operators were considered to generate new individuals, independently, based on the Gaussian and the Cauchy distribution, respectively. The use of the latter distribution has a higher probability of making longer jumps than the former one, due to its long flat tails. Differential Evolution, Chaotic and hybrid mutation strategies have been considered to optimize the performance of BSO [7, 14, 32, 34] by avoiding premature convergence. In [8, 21], the predator-prey method has been proposed for better utilization of the global information of the swarm and diversification of the population. This method considers that the cluster centers play the role of predators, whereas the other solutions play the role of preys. Other approaches that have been proposed to maintain the population diversity include the Niche approach [38], the multiple partial re-initializations [6], and the Max-fitness Clustering Method (MCM) [13]. The latter is used to divide the solutions into sub-groups and obtain multiple global and local optima, in accordance with a self-adaptive parameter control. The control aims to adjust the exploration and exploitation, by reducing similar solutions in subpopulations. In [25] an objective space was used to reduce the computation time for convergence, instead of the solution space. A consequence of this approach is that the computation time becomes dependent on the size of the population, and not on the dimension of the problem. Multi-Objective Differential Brain Storm Optimization (MDBSO) algorithm [33] has been proposed as an extension of this approach using a differential mutation operation instead of the Gaussian. Global-best BSO (GBSO) uses the global-best idea for updating the population [11].An elitist learning strategy of BSO has been proposed in [29]. According to this approach the first half individuals with better fitness values are maintained, while other individuals with worse fitness values can improve their performances by learning from the excellent ones. Cao *et al.* proposed a random grouping strategy as a replacement of the *k*-means clustering method [2], whereas Guo *et al.* proposed a self-adaptive Multiobjective BSO [12]. The combination of the information of one or more clusters has been considered in [5], using affinity propagation, which does not require to know in advance, the number of clusters. In [10] a stagnation-triggered re-initialization scheme has been proposed, where the search space information has been incorporated into the step size update. Agglomerative hierarchical clustering has been considered for BSO in [4], in order to avoid the use of a predefined number of clusters for the grouping of the generated solutions, and to enhance solution searching. In addition, an improved BSO (IBSO) algorithm [28] based on graph theory has been introduced, in order to enhance the diversity of the algorithm and help BSO escape from local optima. A GPU-based implementation of BSO using NVIDIA's CUDA technology has been investigated in [17]. In [18], an objective space-based cluster Multi-objective Brainstorm Optimization algorithm (MOBSO-OS), has been introduced for improving the computational efficiency, considering sparsity and measurement error as two competing cost function terms. The modification of BSO described in [31] is based on an orthogonal experimental design strategy, which aims to discover useful search experiences for improving the convergence and solution accuracy. The convergence of the BSO has also been analyzed by using the Markov model [39].

The termination of the original BSO algorithm and most of its current modifications usually depend on a predefined upper bound of algorithm iterations, whereas the role of clustering in the algorithm convergence has not been sufficiently investigated. To address these issues, in this paper we propose a novel modification of BSO, which considers a differentiated brainstorming scenario from the scenario considered in the original BSO. Specifically, it considers that during the brainstorming process, participating groups with similar ideas agree about the similarity of their ideas and collaborate for the determination of a better solution. This is implemented by following a cluster merging strategy per algorithm iteration, where the most similar clusters represent the groups with the similar ideas. This way, and by employing an elitist approach to the selection of the ideas, the algorithm is directed to convergence. In that sense, the proposed modification of BSO is called Determinative BSO (DBSO).

The rest of this paper is organized in four sections. Section 2 describes the principles of the original BSO, and Sect. 3 presents the proposed DBSO. The experiments performed and the results obtained are presented in Sect. 4. The conclusions of our study are summarized in Sect. 5.

2 Original BSO

Swarm intelligence algorithms have been inspired by the collective behavior of animals like ants, fish, birds, bees, etc. However, BSO is inspired by the most intelligent creature in the world, human being, and the way people brainstorm to find solutions to problems [24]. A facilitator, a brainstorming group of people and several problem owners are necessary to carry out the procedure. Moreover, in order to generate ideas and avoid inhibitions, Osborn's original four rules have to be obeyed. These rules are: 1) No judgment and evaluation should occur during the session; 2) Encourage the creativity of the members; 3) Combination of ideas and improvement are sought. Participants should suggest how ideas of other members can be improved; or how two or more ideas can be combined to create a new idea; 4) Go for quantity. Members should generate as many ideas as possible, because brainstorming mainly focuses on quantity of ideas, rather than their quality. The original BSO, in general, consists of the following four steps: initialization, grouping, generation and selection of solutions. All of them, except the first one, are repeated in each iteration, until a termination condition is met. More specifically, in the beginning, a population of N individuals is generated. Each of these individuals represents a different idea, randomly initialized within a search parameter space. Then, BSO evaluates these ideas according to a fitness function and uses the k-means clustering algorithm to group them into M clusters. The best idea in each cluster is recorded as the cluster center. A partial re-initialization is performed, as a randomly selected center is replaced by a new idea. In order to generate new individuals, BSO in a random way, in the beginning, chooses one or two clusters and then the cluster center that is selected is the one of them all that has higher priority or another idea in the cluster. The new individual generation is updated according to the formula:

$$X_{new}^{d} = X_{sel}^{d} + \xi \cdot n(\mu, \sigma) \tag{1}$$

where X_{new}^{d} is the d^{th} dimension of individual newly generated, $n(\mu, \sigma)$ is the Gaussian random value with mean μ and standard deviation σ, X_{sel}^{d} is the d^{th} dimension of individual selected to generate new individual and ξ is the step-size coefficient, which is a parameter controlling the convergence speed. Step-size is estimated as:

$$\xi = logsig\left(\frac{0.5 \cdot max_{it} - curr_{it}}{k}\right) \cdot p \tag{2}$$

where *logsig* is a logarithmic sigmoid transfer function, k adjusts the slope of the function, max_{it} and $curr_{it}$ denote the maximum number of iterations and current iteration number respectively. Variable p returns a random value within the range (0, 1).

After the idea generation, the newly generated individual is compared with the existing individual, they are evaluated and the better one is kept and recorded as the new individual.

3 Determinative BSO

In the brainstorming process, a group of individuals gather and exchange their ideas, in order to find a solution for a given problem. During the brainstorming, there are many ideas produced by individuals of different background and as the process progresses, possible solutions are discussed and combined, in order to determine the best solution. However, it is common for a human brainstorming process not to be productive [26]. In this paper we consider a scenario where the brainstorming is performed by individuals that are willingly collaborate in order to converge faster to optimal solutions. To this end, individuals who have similar ideas, recognize and agree that their ideas are similar to each other, and they are grouped together to collaborate for the determination of an even better idea. This determinative BSO (DBSO) process is implemented by employing a cluster merging strategy in the grouping of the ideas and by selecting the best ideas per algorithm iteration. In addition, cluster merging is a method to identify general-shaped clusters [3].

Examining the process of the original BSO, described in the previous section, it can be observed that there is not any kind of directionality in the brainstorming process. Moreover, the BSO algorithm needs to be improved in the ability of preventing premature convergence and "jumping out" of local optima. In this paper, we propose a modification of BSO capable of automatically converging to optimal solutions. Particularly, clustering starts with a relatively large number of clusters, which represents possible ideas and whenever clusters are identified, the clusters are merged, after taking into consideration the criteria mentioned above. After the number of clusters has been reduced and there are not any other, different enough, possible solutions, the algorithm ends.

The detailed procedure of the DBSO is presented in Fig. 1, along with the basic steps of the original BSO [24]. The particular steps introduced in DBSO are highlighted with a red dashed line. In **step 1** the initialization of the parameters and the random generation of N individuals (potential solutions), are performed. In **step 2**, the clustering strategy separates the N individuals into $M < N$ clusters, using k-means. In **step 3**, the

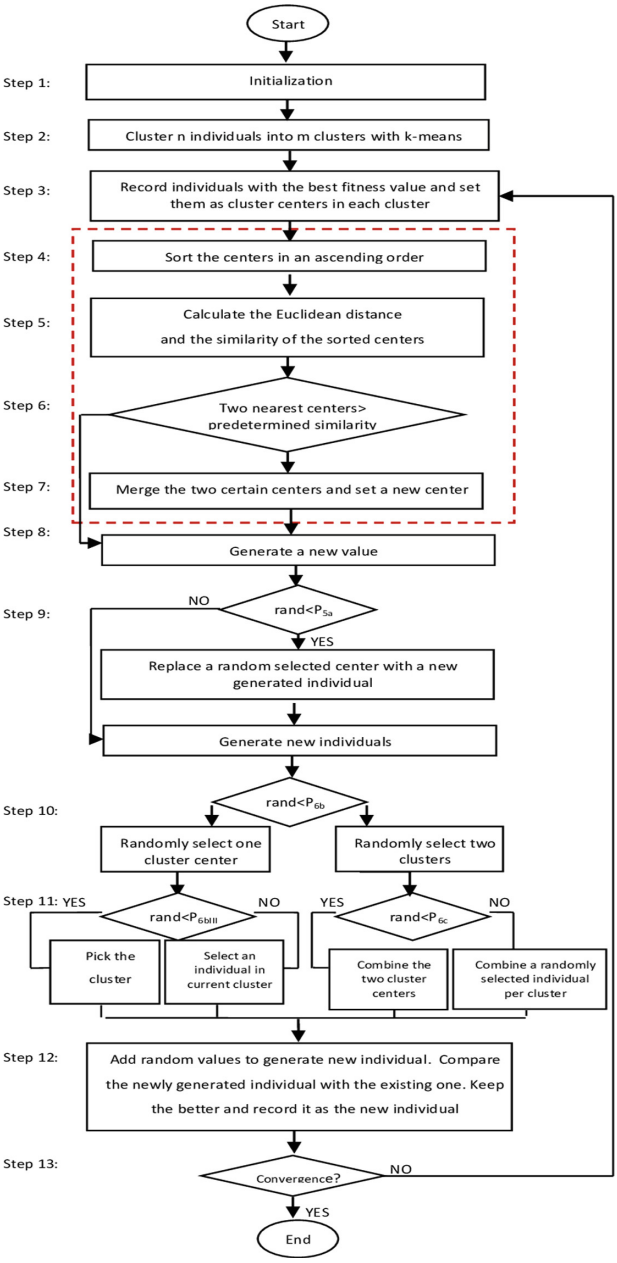

Fig. 1. Flowchart of DBSO algorithm

best individual is recorded as cluster center of each cluster, according to their fitness value. In **step 4**, the cluster centers are sorted in an ascending order. In **step 5**, the Euclidean distance of the centers and then the similarity, are calculated. The similarity S is calculated based on the follwing formula [23]:

$$S = \frac{1}{1 + dist(x, y)} \tag{3}$$

where *dist* is the Euclidean distance between two elements x, y. In **step 6**, the first two sorted centers are selected. In particular, the selected centers are the two centers with the smallest distance among all the sorted centers, which have also the biggest similarity, taking into consideration Eq. (3).

If they are similar enough, in **step 7**, merge the respective cluster centers and set as their new center, the center, which precedes in the ascending order between the two of them. Otherwise, go directly to step 8. In **step 8**, generate a new random value in the range [0, 1); In **step 9**, if the randomly generated value is smaller than a predetermined probability $P_{5a,}$ randomly select a cluster center and then randomly generate an individual to replace the selected center. Then, generate new individuals, otherwise, generate directly new individuals. In **step 10**, randomly select one cluster, with probability P_{6b}, otherwise, select two clusters. In **step 11a**, for the case of the selection of one cluster, of step 10: if a random value generated is smaller than $P_{6b3,}$ which is a probability to select the center of one selected cluster, pick the cluster center. Otherwise, randomly select an individual in the current cluster. Else, for the case of the selection of two clusters, of step 10, in **step 11b**, if a random value generated is smaller than $P_{6c,}$ which is the probability to select the centers of two selected clusters, combine the two cluster centers. Otherwise, two individuals from each selected cluster are randomly selected to be combined. In **step 12**, add random values to the selected centers or individuals, in order to generate a new individual. Then, compare the newly generated individual with the existing ones. Keep the better one and record it as the new individual. In **step 13,** if there is no convergence to the best solution or a termination condition, repeat the algorithm from step 3, until there are not any other similar enough clusters that can be merged. Otherwise, end the algorithm.

4 Experiments and Results

4.1 Parameter Settings and Benchmark Functions

To evaluate DBSO a set of eleven benchmark functions were used, which are presented in Table 1, along with their bounds [16]. DBSO is compared not only with the original BSO, but also with a modified version, named IBSO [28]. For each benchmark function, both DBSO and BSO were executed 50 times, in order to obtain justifiable statistical results, as for different runs and values of parameters different results may be generated. Each of these functions has twenty independent variables, the population was set to be 50 individuals, the maximum number of generations were set to 1000 and the number of clusters 5. Moreover, the similarity degree for the clusters to be merged, which is referred in step 7 of the algorithm, was set to 65%, after preliminary experimentations,

Table 1. Benchmark Functions.

Function	Name	Bounds
f1	Alpine	[−10, 10]
f2	Dixon & Price	[−10, 10]
f3	Griewank	[−100, 100]
f4	Pathological function	[−100, 100]
f5	Schwefel 2.22	[−100,100]
f6	Schwefel 2.26	[−512, 512]
f7	Schwefel 2.21	[−100, 100]
f8	Zakharov	[−5,10]
f9	Ackley	[−35, 35]
f10	Matyas	[−10, 10]
f11	Sphere	[0, 10]

which indicated that it provides best results in most cases. In addition, DBSO was tested with a wider population in order to examine its performance.

The results of the experiments and the comparisons performed are summarized in Table 2, 3 and 4. The best results are indicated in boldface typesetting. The tables include the average values and the standard deviation (±) of the eleven benchmark functions that are presented below, obtained by DBSO in comparison with BSO and IBSO. The number of iterations of DBSO and BSO, in Table 2 and Table 3, shows the number of iterations that are needed in order the algorithm to converge.

4.2 Comparison of DBSO and BSO

In order to investigate the performance of DBSO and BSO on solving several types of problems, the two algorithms were tested on several benchmark functions, introduced in Table 1, with a wider number of population and clusters. Firstly, the experiments were done for a population of 50 and 5 clusters and then for a population of 500 and 10 clusters.

Table 1 includes functions that are multimodal and unimodal. Multimodal are the functions that have multiple local minima. As it can be noticed in Table 2, DBSO performs better than BSO, both in function minimization and the number of iterations required to converge, in multimodal functions (*f1),* (*f6*), (*f9*) and (*f11*). The presented results have been obtained by averaging, over 50 independent runs of the respective algorithms. For the multimodal functions (*f3*) and (*f8*) DBSO perform equivalently to the original BSO. For the unimodal benchmark functions (*f2*), (*f5*), (*f7*), (*f10)* DBSO exhibits also a better performance comparing to BSO. However, in (*f4*) BSO has provided better results than DBSO. As it can be noticed, DBSO converges earlier in most cases, taking into consideration the comparison with the original BSO. Furthermore, the convergence of

DBSO and BSO is illustrated indicatively on Fig. 2. The diagrams have been obtained by averaging the results of 50 independent runs of the respective algorithms.

Moreover, for the case of DBSO with a larger population and number of clusters, the similarity was set to be 60%. The results of the experiments are summarized in Table 3 and, indicatively, the convergence process is presented in Fig. 3. The presented results have been obtained by averaging, over 50 independent runs of the respective algorithms. DBSO, in this case, has a better performance in minimization on four out of four benchmark functions, in comparison with the original BSO. Moreover, considering the number of iterations required for convergence, DBSO converges in less iterations in (*f4*), (*f7*), (*f11*). However, in (*f6*), the required iterations of convergence for BSO seem to be less than those of DBSO. DBSO with a population of 500 individuals and 10 clusters appears to be more stable than DBSO with a population size equal to 50 and 5 clusters. In addition, the convergence is earlier in the latter case of Table 4 in comparison with the respective results presented in Table 2.

Table 2. Results of DBSO and BSO for a population of 50 individuals and 5 clusters.

Function	DBSO	BSO	Iterations of DBSO	Iterations of BSO
f1	**2.4E−01 ± 2.5E−01**	2.9E−01 ± 2.4E−01	**750**	754
f2	**7.4E−01 ± 1.2E−01**	7.9E−01 ± 2.1E−01	**465**	583
f3	**1.6E−01 ± 3.9E−01**	1.6E−01 ± 4.5E−01	**314**	395
f4	7E+00 ± 4.8E−01	**6.6E+00 ± 4.9E−01**	644	**543**
f5	**3.7E−03 ± 1.1E−02**	1.4E−02 ± 6.1E−02	**744**	890
f6	**3E+03 ± 6.5E+02**	3.3E+03 ± 834.8E+00	**256**	279
f7	**7.98E−03 ± 2E−02**	8.21E03 ± 7.4E−03	**740**	952
f8	**1.9E−01 ± 8.7E−01**	1.9E−01 ± 1.4E−01	**711**	831
f9	**6.9E−13 ± 4E−13**	8.3E-13 ± 4.1E−13	**879**	951
f10	**1E−26 ± 1E−26**	1.1E−26 ± 1.3E−26	**974**	990
f11	**2.8E−21 ± 9.7E−22**	2.9E−21 ± 9.7E−22	**94**	99

Table 3. Comparison of DBSO and BSO for a population of 500 individuals and 10 clusters.

Function	DBSO	BSO	Iterations of DBSO	Iterations of BSO
f4	**5.9E+00 ± 1.4E−01**	6.1E+00 ± 1.8E−01	**525**	620
f6	**2.8E+03 ± 5.7E+02**	2.9E+03 ± 6.7E+02	165	**116**
f7	**4E−15 ± 6.1E−16**	1.5E−11 ± 1.7E-12	**110**	126
f11	**5.8E−29 ± 1E−29**	8.6E−22 ± 1.7E−22	**110**	115

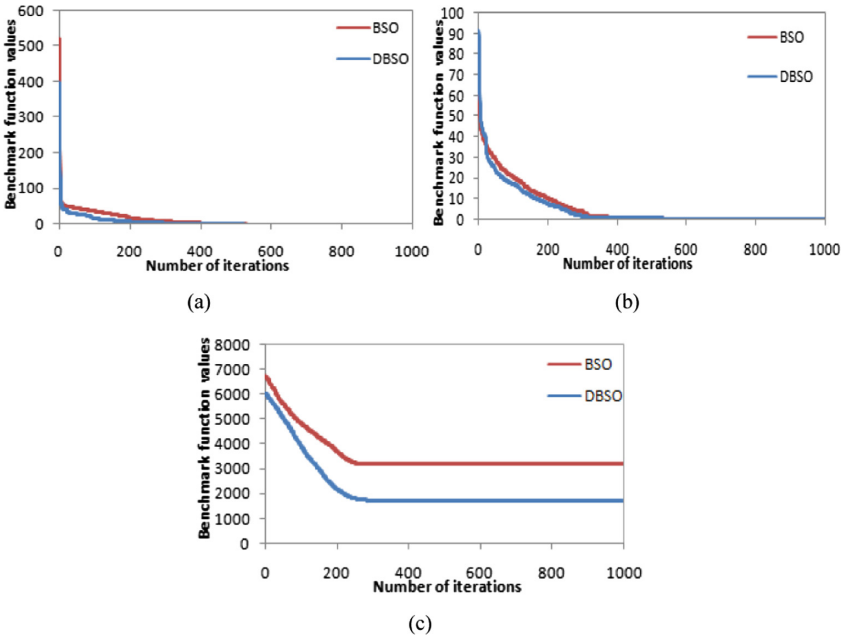

Fig. 2. Convergence of DBSO and BSO tested on benchmark functions for a population of 50 individuals and 5 clusters; (a) Dixon & Price, (b) Schwefel 2.21, (c) Schwefel 2.26.

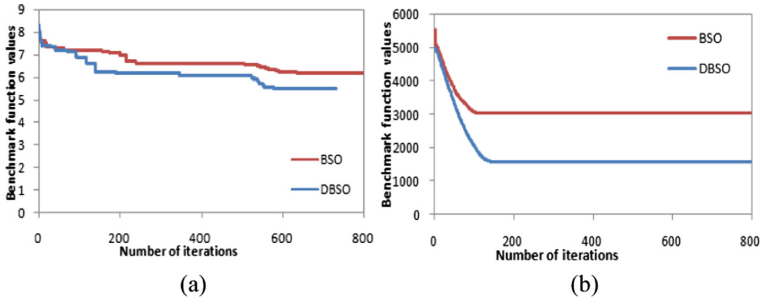

Fig. 3. Convergence of DBSO and BSO tested on benchmark functions for a population of 500 individuals and 10 clusters. (a) Pathological function, (b) Schwefel 2.26.

4.3 Comparison of DBSO and IBSO

The proposed DBSO algorithm was compared also with the state-of-the-art algorithm IBSO, which is a modified version of BSO based on graph theory [28]. The parameters of DBSO were set to the same values with those used in [28], for a fair comparison. The results of the comparison of DBSO and IBSO are presented in Table 4. As it can be observed from Table 4, DBSO has better results on six out of seven benchmark functions (*f1–f3*) and (*f5–f7*). IBSO, though, provides better results comparing to DBSO in (*f4*).

Table 4. DBSO and BSO tested on 5 benchmark functions

Function	DBSO	IBSO
f1	**2.4E−01 ± 2.5E−01**	3.5E−01 ± 3.1E−01
f2	**7.4E−01 ± 1.2E−01**	3.7E+00 ± 8.6E+00
f3	**1.6E−01 ± 3.9E−01**	1E+00 ± 2.4E−16
f4	7E+00 ± 4.8E−01	**6.7E+00 ± 4.7E−01**
f5	**3.7E−03 ± 1.1E−02**	6.4E−03 ± 1E−02

5 Discussion and Conclusions

In this paper, a new improved version of BSO is introduced, named DBSO. DBSO has been modified in order to converge more efficiently and effectively to optimal solutions. Specifically, the proposed algorithm considers that during the brainstorming process, participating groups with similar ideas agree about the similarity of their ideas and collaborate for the determination of a better solution. This is achieved by introducing a cluster merging strategy, per algorithm iteration, where the most similar clusters represent the groups with the similar ideas. Along with the elitist approach followed to the selection of the ideas, the proposed BSO modification provides a faster convergence compared to current BSO algorithms, while maintaining the diversity among the fittest solutions.

According to the results of the experiments and the comparisons performed:

- DBSO performs well both in multimodal and unimodal functions;
- The proposed method helps the algorithm find the best solution, while forces the algorithm converge in less iterations;
- DBSO succeeds a satisfactory stability in repeated experiments, after directing the solutions to the better one.

BSO possesses a great potential as an optimization tool and it has already been applied to solve successfully problems in various domains. It is a promising algorithm; however, it needs further investigation. With regard to future work, we plan to improve and investigate DBSO even more, addressing open questions related, but not limited to, the following:

- More benchmark functions with larger dimensions and wider population sizes;
- The development of hybrid models based on BSO for complicated real-time optimization problems;
- Comparisons of DBSO with other swarm intelligence algorithms, including other versions of BSO, e.g., chaotic, and state-of-the-art nature inspired approaches.

Acknowledgment. This research has been co-financed by the European Union and Greek national funds through the Operational Program Competitiveness, Entrepreneurship and Innovation, under the call RESEARCH—CREATE—INNOVATE (project code: T1EDK-02070).

References

1. Anbarasi, M., Saleem Durai, M.: Prediction of protein folding kinetics states using hybrid brainstorm optimization. Int. J. Comput. Appl. **2018**, 1–9 (2018)
2. Cao, Z., Shi, Y., Rong, X., Liu, B., Du, Z., Yang, B.: Random grouping brain storm optimization algorithm with a new dynamically changing step size. In: Tan, Y., Shi, Y., Buarque, F., Gelbukh, A., Das, S., Engelbrecht, A. (eds.) ICSI 2015. LNCS, vol. 9140, pp. 357–364. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-20466-6_38
3. Celebi, M.E., et al.: A comparative study of efficient initialization methods for the k-means clustering algorithm. Expert Syst. Appl. **40**(1), 200–210 (2013)
4. Chen, J., Wang, J., Cheng, S., Shi, Y.: Brain storm optimization with agglomerative hierarchical clustering analysis. In: Tan, Y., Shi, Y., Li, L. (eds.) ICSI 2016. LNCS, vol. 9713, pp. 115–122. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-41009-8_12
5. Chen, J., Cheng, S., Chen, Y., Xie, Y., Shi, Y.: Enhanced brain storm optimization algorithm for wireless sensor networks deployment. In: Tan, Y., Shi, Y., Buarque, F., Gelbukh, A., Das, S., Engelbrecht, A. (eds.) ICSI 2015. LNCS, vol. 9140, pp. 373–381. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-20466-6_40
6. Cheng, S., et al.: Population diversity maintenance in brain storm optimization algorithm. J. Artif. Intell. Soft Comput. Res. **4**(2), 83–97 (2014)
7. Chu, X., Chen, J., Cai, F., Chen, C., Niu, B.: Augmented brain storm optimization with mutation strategies. In: Shi, Y., et al. (eds.) SEAL 2017. LNCS, vol. 10593, pp. 949–959. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-68759-9_78
8. Duan, H., et al.: Predator–prey brain storm optimization for DC brushless motor. IEEE Trans. Magn. **49**(10), 5336–5340 (2013)
9. Duan, H., Li, C.: Quantum-behaved brain storm optimization approach to solving Loney's solenoid problem. IEEE Trans. Magn. **51**(1), 1–7 (2015)
10. El-Abd, M.: Brain storm optimization algorithm with re-initialized ideas and adaptive step size. In: 2016 IEEE Congress on Evolutionary Computation (CEC 2016), pp. 2682–2686 (2016)
11. El-Abd, M.: Global-best brain storm optimization algorithm. Swarm Evol. Comput. **37**(2017), 27–44 (2017)
12. Guo, X., Wu, Y., Xie, L., Cheng, S., Xin, J.: An adaptive brain storm optimization algorithm for multiobjective optimization problems. In: Tan, Y., Shi, Y., Buarque, F., Gelbukh, A., Das, S., Engelbrecht, A. (eds.) ICSI 2015. LNCS, vol. 9140, pp. 365–372. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-20466-6_39
13. Guo, X., Wu, Y., Xie, L.: Modified brain storm optimization algorithm for multimodal optimization. In: Tan, Y., Shi, Y., Coello, C.A.C. (eds.) ICSI 2014. LNCS, vol. 8795, pp. 340–351. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-11897-0_40
14. Guo, Y., et al.: Grid-based dynamic robust multi-objective brain storm optimization algorithm. Soft. Comput. **2019**, 1–21 (2019). https://doi.org/10.1007/s00500-019-04365-w
15. Ibrahim, R.A., et al.: Galaxy images classification using hybrid brain storm optimization with moth flame optimization. J. Astron. Telescopes Instrum. Syst. **4**(3), 038001 (2018)
16. Jamil, M., Yang, X.-S.: A literature survey of benchmark functions for global optimization problems. arXiv preprint arXiv:1308.4008 (2013)

17. Jin, C., Qin, A.K.: A GPU-based implementation of brain storm optimization. In: 2017 IEEE Congress on Evolutionary Computation (CEC 2017), pp. 2698–2705 (2017)
18. Liang, J., et al.: Multi-objective brainstorm optimization algorithm for sparse optimization. In: 2018 IEEE Congress on Evolutionary Computation (CEC 2018), pp. 1–8 (2018)
19. Madheswari, K., et al.: Visible and thermal image fusion using curvelet transform and brain storm optimization. In: Region 10 Conference (TENCON), 2016 IEEE (2016), pp. 2826–2829 (2016)
20. Osborn, A.F.: Applied Imagination, Scribner. Charles Scribner, New York (1953)
21. Qiu, H., et al.: Chaotic predator-prey brain storm optimization for continuous optimization problems. In: 2017 IEEE Symposium Series on Computational Intelligence (SSCI 2017), pp. 1–7 (2017)
22. Ramadan, R., Khedr, A.: Brain storming algorithm for coverage and connectivity problem in wireless sensor network. In: Communication, Management and Information Technology: International Conference on Communciation, Management and Information Technology, Cosenza, Italy, 26–29 April 2016 (ICCMIT 2016), p. 371 (2016)
23. Segaran, T.: Collective Intelligence-Building Smart Web 2.0 Applications. O'Reilly, Newton (2007)
24. Shi, Y.: Brain storm optimization algorithm. In: Tan, Y., Shi, Y., Chai, Y., Wang, G. (eds.) ICSI 2011. LNCS, vol. 6728, pp. 303–309. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-21515-5_36
25. Shi, Y.: Brain storm optimization algorithm in objective space. In: 2015 IEEE Congress on evolutionary computation (CEC 2015), 1227–1234 (2015)
26. Stroebe, W., et al.: Beyond productivity loss in brainstorming groups: the evolution of a question. Adv. Exp. Soc. Psychol. **43**, 157–203 (2010)
27. Sun, C., et al.: Optimal satellite formation reconfiguration based on closed-loop brain storm optimization. IEEE Comput. Intell. Mag. **8**(4), 39–51 (2013)
28. Wang, G.-G., et al.: An improved brain storm optimization algorithm based on graph theory. In: 2017 IEEE Congress on Evolutionary Computation (CEC 2017), pp. 509–515 (2017)
29. Wang, H., Liu, J., Yi, W., Niu, B., Baek, J.: An improved brain storm optimization with learning strategy. In: Tan, Y., Takagi, H., Shi, Y. (eds.) ICSI 2017. LNCS, vol. 10385, pp. 511–518. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-61824-1_56
30. Wang, J., et al.: Improved v-support vector regression model based on variable selection and brain storm optimization for stock price forecasting. Appl. Soft Comput. **49**(2016), 164–178 (2016)
31. Wang, R., et al.: Brain storm optimization algorithm based on improved clustering approach using orthogonal experimental design. In: 2019 IEEE Congress on Evolutionary Computation (CEC 2019), pp. 262–270 (2019)
32. Wu, Y., Xie, L., Liu, Q.: Multi-objective brain storm optimization based on estimating in knee region and clustering in objective-space. In: Tan, Y., Shi, Y., Niu, B. (eds.) Advances in Swarm Intelligence. ICSI 2016. LNCS, vol. 9712, pp. 479–490. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-41000-5_48
33. Wu, Y., Wang, X., Xu, Y., Fu, Y.: Multi-objective differential-based brain storm optimization for environmental economic dispatch problem. In: Cheng, S., Shi, Y. (eds.) Brain Storm Optimization Algorithms. ALO, vol. 23, pp. 79–104. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-15070-9_4
34. Xie, L., Wu, Y.: A modified multi-objective optimization based on brain storm optimization algorithm. In: Tan, Y., Shi, Y., Coello, C.A.C. (eds.) ICSI 2014. LNCS, vol. 8795, pp. 328–339. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-11897-0_39
35. Xue, J., Wu, Y., Shi, Y., Cheng, S.: Brain storm optimization algorithm for multi-objective optimization problems. In: Tan, Y., Shi, Y., Ji, Z. (eds.) ICSI 2012. LNCS, vol. 7331, pp. 513–519. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-30976-2_62

36. Zhan, Z., et al.: A modified brain storm optimization. In: 2012 IEEE Congress on Evolutionary Computation (CEC 2012), pp. 1–8 (2012)
37. Zhou, D., Shi, Y., Cheng, S.: Brain storm optimization algorithm with modified step-size and individual generation. In: Tan, Y., Shi, Y., Ji, Z. (eds.) ICSI 2012. LNCS, vol. 7331, pp. 243–252. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-30976-2_29
38. Zhou, H.J., et al.: Niche brain storm optimization algorithm for multi-peak function optimization. Adv. Mater. Res. **989**, 1626–1630 (2014)
39. Zhou, Z., et al.: Convergence analysis of brain storm optimization algorithm. In: 2016 IEEE Congress on Evolutionary Computation (CEC 2016), pp. 3747–3752 (2016)