Soft Computing (2019) 23:715–734
https://doi.org/10.1007/s00500-018-3102-4

FOUNDATIONS

Butterfly optimization algorithm: a novel approach for global optimization

Sankalap Arora[1] · Satvir Singh[1]

Published online: 8 March 2018

Abstract
Real-world problems are complex as they are multidimensional and multimodal in nature that encourages computer scientists to develop better and efficient problem-solving methods. Nature-inspired metaheuristics have shown better performances than that of traditional approaches. Till date, researchers have presented and experimented with various nature-inspired metaheuristic algorithms to handle various search problems. This paper introduces a new nature-inspired algorithm, namely butterfly optimization algorithm (BOA) that mimics food search and mating behavior of butterflies, to solve global optimization problems. The framework is mainly based on the foraging strategy of butterflies, which utilize their sense of smell to determine the location of nectar or mating partner. In this paper, the proposed algorithm is tested and validated on a set of 30 benchmark test functions and its performance is compared with other metaheuristic algorithms. BOA is also employed to solve three classical engineering problems (spring design, welded beam design, and gear train design). Results indicate that the proposed BOA is more efficient than other metaheuristic algorithms.

Keywords Butterfly optimization algorithm · Global optimization · Nature inspired · Metaheuristic · Benchmark test functions · Engineering design problems

1 Introduction

For million of years, nature has been developing many biological systems and helping them in their survival. With the time, these natural systems have become so robust and efficient that they can solve most of the real-world problems (Fister et al. 2013). Using key characteristics of these biological systems, various metaheuristic algorithms have been developed and employed to various optimization problems, whereas the conventional optimization algorithms fail to produce satisfactory results for problems with nonlinearity and multimodality (Back 1996; Onwubolu and Babu 2004). In engineering, many design applications require optimal solution under highly complex constraints over a short duration of time which is a very challenging task (Brownlee 2011) and these metaheuristic algorithms are known to have intriguing efficiency for solving complex and multidimensional problems in a shorter period of time (Yang 2010a). Till now, several metaheuristic algorithms have been investigated to solve real-world combinatorial or global optimization problems such as Artificial Bee Colony (ABC) (Karaboga and Basturk 2007), Cuckoo Search (CS) (Yang and Deb 2009), Differential Evolution (DE) (Storn and Price 1997), Firefly Algorithm (FA) (Yang 2010b), Genetic Algorithm (GA) (Goldberg and Holland 1988), Monarch Butterfly Optimization (MBO) (Wang et al. 2015) and Particle Swarm Optimization (PSO) (Eberhart and Shi 2001). These algorithms demonstrate improved performances when compared with conventional optimization techniques, especially when applied to solve non-convex optimization problems (Talbi 2009). Up to now, researchers have only used very limited characteristics inspired by nature and there is room for more algorithm development (Yang 2010a; Onwubolu and Babu 2004; Wolpert and Macready 1997).

In this paper, a new nature-inspired metaheuristic algorithm for global optimization named as Butterfly Opti-

Communicated by A. Di Nola.

✉ Sankalap Arora
sankalap.arora@gmail.com

Satvir Singh
DrSatvir.in@gmail.com

[1] I. K. GUJRAL Punjab Technical University, Jalandhar, Punjab, India

mization Algorithm (BOA in short) which is inspired by the foraging behavior of the butterflies is proposed. The behavior of butterflies can be described as their cooperative movement toward the food source position. The butterflies receive/sense and analyze the smell in the air to determine the potential direction of a food source/mating partner. BOA mimics this behavior to find the optima in the hyper search space. The contribution of this paper is that a novel nature-inspired metaheuristic algorithm is presented and investigated. This population-based general-purpose metaheuristic demonstrates outstanding performance on benchmark test functions as well as on classical engineering design problems.

After this review, the rest of this paper is organized as follows. First, some related work on nature-inspired metaheuristic algorithms is presented in Sect. 2. Section 3 discusses the biological behavior of butterflies and illustrates BOA framework so as to use it as an optimization search algorithm. This section also discusses the conceptual comparison of proposed BOA with other optimization algorithms. To test and validate this new algorithm, a testbed of 30 benchmark optimization functions (problems) is considered and presented in Sect. 4. Section 5 presents simulation results and performance analysis of BOA as the algorithm is subjected to benchmark test function under various constraints. To evaluate the performance of the proposed BOA over engineering problems, vibration-based damage detection problems in three scenarios (spring design, welded beam design, and gear train design) have been solved and results are discussed in Sect. 6. Finally, this paper is concluded and future research work is proposed in Sect. 7.

2 Related work

Metaheuristic algorithms mimic natural phenomenon in order to search for an optimal solution. Among the various metaheuristic algorithms, ABC, CS, DE, FA, GA, MBO, and PSO are widely known algorithms.

ABC is based on the intelligent behavior of honey bee swarm which categorizes the bees in a hive into three types viz. scout bees, employed bees and onlooker bees. Scout bees fly randomly without any guidance. Employed bees search the neighborhood of their positions, and they share the information of these food sources to the onlooker bees. Onlooker bees use fitness of the population to select a guiding solution for exploitation. These onlooker bees tend to select good food sources from those found by the employed bees. The food source that has higher quality (fitness) will have a large chance to be selected by the onlooker bees than one of lower quality (Karaboga and Basturk 2007). The scout bees are translated from a few employed bees, which abandon their food sources and search new ones. The algorithm is mainly used for multimodal function optimization (Karaboga and Basturk 2008).

CS is a metaheuristic algorithm based on the obligate brood parasitic behavior of some cuckoo species in which the cuckoo bird lay their eggs in the nests of birds of different species (Gandomi et al. 2013a). Each solution is represented by an egg and a new solution is represented by a cuckoo egg. The algorithm's basic methodology is based on replacing a not-so-good solution by new or potentially better solutions. Cuckoo search idealized such breeding behavior to perform optimization (Arora and Singh 2013b).

DE is a simple yet powerful population-based stochastic search technique which uses vector differences for perturbing the vector population and it maintains a population of candidate solutions subjected to iterations of recombination, evaluation, and selection (Storn and Price 1997). The recombination approach facilitates the creation of new solutions depending upon the weighted difference between two randomly selected population members added to a third population member. This perturbs population members relative to the spread of the broader population. Additionally, with the selection, the perturbation effect self-organizes the sampling of the hyper search space, bounding it to known areas of interest.

In FA, the social flashing behavior of fireflies is modeled in algorithmic form to search optimal solution to arbitrary problems (Arora and Singh 2013a). The underlying mechanism is that one firefly gets attracted to other fireflies' specific flashing pattern which is produced by a process of bioluminescence. The intensity of the flash is directly proportional to the fitness of the individual (Gupta and Arora 2015). FA uses the concept of attraction where the less bright one attracts to more brighter one which allows individuals to change their location from one place to another. This movement enables the swarm to explore the optimum solution over the search space (Yang 2009; Arora et al. 2014).

In GA, a problem solution is considered as the individual's chromosome and a population of such individuals strives to survive under harsh conditions (Holland 1992; Goldberg and Holland 1988). GA is based on the Darwinian theory of survival of the fittest. GA is based on three operators: selection, crossover and mutation (Goldberg and Holland 1988). In selection, highly fitted individuals are selected to generated descendants, which are used by the crossover operator. In mutation, a location inside the chromosome is selected and its value is changed depending on the chromosome encoding used.

MBO is based on the southward migration of the eastern North American monarch population to Mexico. In this algorithm, the population of butterfly individuals is divided into two different lands viz. southern Canada and the northern USA (Land 1) and Mexico (Land 2). The location of butterfly individuals is updated by using two operators, viz.

migration operator and butterfly adjusting operator. The role of migration operator is to create new offsprings while butterfly adjusting operator is used for tuning the positions for other butterfly individuals. Migration ratio is the deciding factor for the migration operator which generates new offsprings of monarch butterflies (Wang et al. 2015).

PSO algorithm mimics the interaction in a group of species, e.g., fish and birds, etc., for finding the optimal solution in complex search spaces (Eberhart and Shi 2001). The group is represented by a swarm of particles (solutions) and the algorithm described how particles keep moving, as their neighbors find better solutions in the search solution space. Over time, individuals are drawn toward one another's successes, that results in clustering of individuals in optimal regions of the space. PSO uses two kinds of information, (1) cognitive learning, where the particle learns from its own past experiences and (2) social learning, where particle learns from other's experience. The algorithm combines both the learning phenomenon leading the solution to evolve in a better way as the algorithm proceeds (Kennedy 2010).

3 Butterfly optimization algorithm

In this paper, a novel optimization technique is proposed that mimics the food foraging behavior of butterflies. To understand this new algorithm some biological facts and how to model them in BOA are discussed in following subsections.

3.1 Butterfly

In the Linnaean system of Animal Kingdom, butterflies lie in the class of Lepidoptera. There are more than 18,000 species of butterflies across the world. The reason of their survival for so many million years lies in their senses (Saccheri et al. 1998). Butterflies use their sense of smell, sight, taste, touch and hearing to find food and mating partner. These senses are also helpful in migrating from one place to another, escaping from predator and laying eggs in appropriate places. Among all these senses, smell is the most important sense which helps butterfly to find food, usually nectar, even from long distances (Blair and Launer 1997). In order to find the source of nectar, butterflies use sense receptors which are used to smell and these receptors are scattered over butterfly's body parts like antennae, legs, palps, etc. These receptors are actually nerve cells on butterfly's body surface and are called chemoreceptors (Pollard and Yates 1994). These chemoreceptors guide the butterfly to find the best mating partner in order to continue a strong genetic line. A male butterfly is able to identify the female through her pheromone which are scent secretions emitted by the female butterfly to cause specific reactions. Fig. 1 provides some pictures of butterflies.

Based on scientific observations, it is found that butterflies have a very accurate sense of locating the source of fragrance (Raguso 2008). Furthermore, they can separate different fragrances and sense their intensities (Wyatt 2003). Butterflies are search agents of BOA to perform optimization. A butterfly will generate fragrance with some intensity which is correlated with its fitness, i.e., as a butterfly moves from one location to another, its fitness will vary accordingly. The fragrance will propagate over distance and other butterflies can sense it and this is how the butterflies can share its personal information with other butterflies and form a collective social knowledge network. When a butterfly is able to sense fragrance from any other butterfly, it will move toward it and this phase is termed as global search in the proposed algorithm. In another scenario, when a butterfly is not able to sense fragrance from the surrounding, then it will move randomly and this phase is termed as local search in the proposed algorithm. In this paper, terms smell and fragrance are used interchangeably.

3.2 Fragrance

In BOA, each fragrance has its own unique scent and personal touch. It is one of the main characteristics that distinguishes

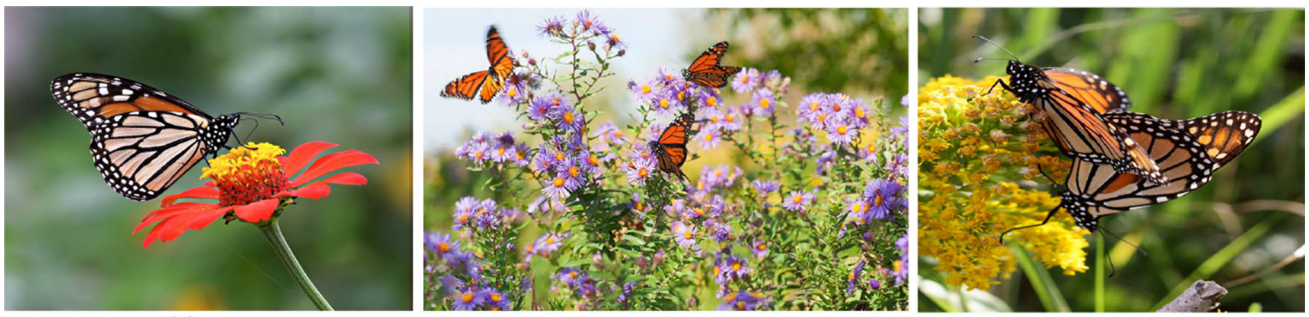

Fig. 1 Social organization and behavior. **a** Butterfly. **b** Food foraging. **c** Butterfly mating among flowers

BOA from other metaheuristics. In order to understand how fragrance is calculated in BOA, first we need to understand, how a modality like smell, sound, light, temperature, etc. is processed by a stimulus. The whole concept of sensing and processing the modality is based on three important terms viz. sensory modality (c), stimulus intensity (I) and power exponent (a). In sensory modality, sensory means to measure the form of energy and process it in similar ways and modality refers to the raw input used by the sensors. Now different modalities can be smell, sound, light, temperature and in BOA, modality is fragrance. I is the magnitude of the physical/actual stimulus. In BOA, I is correlated with the fitness of the butterfly/solution. This means that when a butterfly is emitting a greater amount of fragrance, the other butterflies in that surrounding can sense it and gets attracted toward it. Power is the exponent to which intensity is raised. The parameter a allows for regular expression, linear response and response compression. Response expansion is when I increases, the fragrance (f) increases more quickly than I. Response compression is when I increases, f increases more slowly than I. Linear response is when I increases, f increases proportionally (Baird and Noma 1978; MacKay 1963). Scientists have conducted several experiments on insects, animals and humans for magnitude estimation and they have concluded that sometimes as the stimulus gets stronger, insects become increasingly less sensitive to the stimulus changes (Zwislocki 2009; Stevens 1975). So in BOA, to estimate the magnitude of I, response compression is used.

The natural phenomenon of butterflies is based on two important issues: the variation of I and formulation of f. For simplicity, I of a butterfly is associated with the encoded objective function. However, f is relative, i.e., it should be sensed by other butterflies. According to Steven's power law (Stevens 1975) in order to differentiate smell from other modalities, c is used. Now, as the butterfly with less I moves toward butterfly with more I, f increases more quickly than I. So we should allow f to vary with a degree of absorption which is achieved by the power exponent parameter a. Using these concepts, in BOA, the fragrance is formulated as a function of the physical intensity of stimulus as follows:

$$f = cI^a \tag{1}$$

where f is the perceived magnitude of the fragrance, i.e., how stronger the fragrance is perceived by other butterflies, c is the sensory modality, I is the stimulus intensity and a is the power exponent dependent on modality, which accounts the varying degree of absorption. For most of the cases in our implementation, we can take a and c in the range [0, 1]. The parameter a is the power exponent dependent on modality (fragrance in our case) which means it characterizes the variation of absorption. In one extreme, $a = 1$, this means there is no absorption of fragrance, i.e., the amount of fragrance emitted by a particular butterfly is sensed in the same capacity by the other butterflies. This equivalent to saying that fragrance is propagated in an idealized environment. Thus, a butterfly emitting fragrance can be sensed from anywhere in the domain. Thus, a single (usually global) optimum can be reached easily. On the other hand, if $a = 0$, it means that the fragrance emitted by any butterfly cannot be sensed by the other butterflies at all. So, the parameter a controls the behavior of the algorithm. Another important parameter is c which is also crucial parameter in determining the speed of convergence and how the BOA algorithm behaves. Theoretically $c \in [0, \infty]$ but practically it is determined by the characteristic of the system to be optimized. The values of a and c crucially affect the convergence speed of the algorithm. In the case of maximization problem, the intensity can be proportional to the objective function. Other forms of intensity can be defined in a similar way to the fitness function in firefly algorithm (Yang 2010b), genetic algorithms or the bacterial foraging algorithm (Gazi and Passino 2004).

3.3 Movement of butterflies

To demonstrate above discussions in terms of a search algorithm, the above characteristics of butterflies are idealized as follows:

1. All butterflies are supposed to emit some fragrance which enables the butterflies to attract each other.
2. Every butterfly will move randomly or toward the best butterfly emitting more fragrance.
3. The stimulus intensity of a butterfly is affected or determined by the landscape of the objective function.

There are three phases in BOA:(1) Initialization phase, (2) Iteration phase and (3) Final phase. In each run of BOA, first the initialization phase is executed, then searching is performed in an iterative manner and in the last phase, the algorithm is terminated finally when the best solution is found. In the initialization phase, the algorithm defines the objective function and its solution space. The values for the parameters used in BOA are also assigned. After setting the values, the algorithm proceeds to create an initial population of butterflies for optimization. As the total number of butterflies remains unchanged during the simulation of BOA, a fixed size memory is allocated to store their information. The positions of butterflies are randomly generated in the search space, with their fragrance and fitness values calculated and stored. This finishes the initialization phase and the algorithm starts the iteration phase, which performs the search with the artificial butterflies created.

The second phase of the algorithm, i.e., iteration phase, a number of iterations are performed by the algorithm. In each iteration, all butterflies in solution space move to new positions and then their fitness values are evaluated. The algorithm first calculates the fitness values of all the butterflies on different positions in the solution space. Then these butterflies will generate fragrance at their positions using Eq. (1). There are two key steps in the algorithm, i.e., global search phase and local search phase. In global search phase, the butterfly takes a step toward the fittest butterfly/solution g^* which can be represented using Eq. (2)

$$x_i^{t+1} = x_i^t + \left(r^2 \times g^* - x_i^t\right) \times f_i \tag{2}$$

where x_i^t is the solution vector x_i for ith butterfly in iteration number t. Here, g^* represents the current best solution found among all the solutions in current iteration. Fragrance of ith butterfly is represented by f_i and r is a random number in [0, 1].

Local search phase can be represented as

$$x_i^{t+1} = x_i^t + \left(r^2 \times x_j^t - x_k^t\right) \times f_i \tag{3}$$

where x_j^t and x_k^t are jth and kth butterflies from the solution space. If x_j^t and x_k^t belongs to the same swarm and r is a random number in [0, 1] then Eq. (3) becomes a local random walk.

Search for food and mating partner by butterflies can occur at both local and global scale. Considering physical proximity and various other factors like rain, wind, etc., search for food can have a significant fraction p in an overall mating partner or food searching activities of butterflies. So a switch probability p is used in BOA to switch between common global search to intensive local search.

Till the stopping criteria is not matched, the iteration phase is continued. The stopping criteria can be defined in different ways like maximum CPU time used, maximum iteration number reached, the maximum number of iterations with no improvement, a particular value of error rate is reached or any other appropriate criteria. When the iteration phase is concluded, the algorithm outputs the best solution found with its best fitness. The above-mentioned three steps make up the complete algorithm of butterfly optimization algorithm and its pseudo code is explained in "Algorithm 1".

Algorithm 1 Butterfly optimization algorithm

```
1: Objective function f(x), x=(x_1, x_2, ..., x_dim) , dim = no. of dimensions
2: Generate initial population of n Butterflies x_i= (i=1, 2, ..., n)
3: Stimulus Intensity I_i at x_i is determined by f(x_i)
4: Define sensor modality c, power exponent a and switch probability p
5: while stopping criteria not met do
6:     for each butterfly bf in population do
7:         Calculate fragrance for bf using Eq. (1)
8:     end for
9:     Find the best bf
10:    for each butterfly bf in population do
11:        Generate a random number r from [0, 1]
12:        if r < p then
13:            Move towards best butterfly/solution using Eq. (2)
14:        else
15:            Move randomly using Eq. (3)
16:        end if
17:    end for
18:    Update the value of a
19: end while
20: Output the best solution found.
```

3.4 Conceptual comparison of BOA with other metaheuristics

In the past few decades, various metaheuristic algorithms have been introduced. Among them, ABC, CS, DE, FA, GA, MBO and PSO are most widely studied and employed algorithms. Even though BOA belongs to category of metaheuristic algorithms but it is quite different from previous algorithms.

Originally, PSO was proposed for solving continuous optimization problems. The major difference between BOA and PSO is their different biology backgrounds. PSO was inspired by coordinated group animal of schools of fishes or flock of birds, whereas BOA is inspired by social butterfly foraging strategy, which belongs to the category of the general searching behavior of the social animal. Another important difference between BOA and PSO is information propagation method which is neglected in PSO as each particle is assumed to have awareness of all the information about other particles without any loss. In BOA, information is propagated to all other agents using fragrance. This model of information propagation forms a general knowledge system with some information loss. Even there is no research done yet that how information loss will affect the process of optimization in BOA, but the results in Sect. 5 indicate that this information propagation model contributes to superior performance of BOA over PSO.

Although both BOA and FA draw inspiration from foraging strategy of social animals, but still there are some significant differences. Fireflies produce short and rhythmic flashes of light to attract mating partners and to attract potential preys whereas butterflies use their sense of smell to locate their food. Another important difference is the searching pattern. In FA, every firefly is made to move toward every brighter firefly and some randomness is added in that movement whereas, in BOA, each butterfly moves toward the best butterfly. As far as randomness is concerned, in BOA, a switch probability is used which decides whether the butterfly will move toward the best butterfly or perform a random walk which is not used in FA. So there are significant differences in foraging strategy of butterflies and fireflies.

Another widely used algorithm is GA. BOA and GA are totally different algorithms in terms of inspiration and modeling. GA is based on Darwinian's theory of survival of the fittest, whereas BOA is inspired from social animal foraging strategy of butterflies. The second important difference is that in GA only good solutions have a greater probability of creating new solutions. This means bad solutions will get replaced by new good solutions whereas, in BOA, no solution is discarded. In BOA, every solution is given equal opportunity to improve. The third variation between these two algorithms is generation of new solutions. In GA, two solutions are selected based on their fitness, i.e., greater the fitness, higher the probability to get selected. Then, these solutions contribute in the creation of a new solution and after which the new solution is mutated randomly whereas in BOA, either the butterfly will move toward that butterfly which is emitting more fragrance among all butterflies or it will move randomly and a new solution is created. Switch probability decides whether a butterfly will move toward best butterfly or randomly. So considering the abovementioned factors, it is evident that GA and BOA follow totally different approaches for solving optimization problems.

There is one another algorithm MBO which resembles BOA as these algorithms operate on butterflies but these algorithms do not share the same ideology. In MBO, the location of butterfly individuals is updated by using two operators, i.e., migration operator and butterfly adjusting operator. Migration operator is responsible for creating new offsprings while butterfly adjusting operator plays the critical role of tuning the positions for other butterfly individuals. Migration ratio plays a critical role in the performance of MBO as it controls the migration operator which generates new offsprings of monarch butterflies. In BOA, the butterflies are analyzed from the aspect of food foraging whereas, in MBO, their migratory behavior is considered. BOA considers the ability to sense fragrance among the flowers to locate their food. Moreover, BOA is critically dependent on how the fragrance is emitted, propagated in the environment and sensed by other butterflies in that search space. BOA also considers the fact that fragrance will be absorbed during propagation and this absorption factor cannot be neglected in the actual environment. Another major difference is that MBO uses lèvy flight in position updation of monarch butterflies while BOA uses random number in order to update the location of butterflies. Considering these mentioned factors, it is clear that MBO and BOA do not follow the same ideology and these two algorithms are conceptually very different.

There is another category of metaheuristic algorithms like ABC where the population is structured into different categories. Individuals following in different category perform different tasks and the whole population cooperates to search for the best solution over the solution space. In ABC, the probability of selecting a nectar source is calculated using fitness value of each bee divided by fitness values of all bees. The new position of an onlooker bee is calculated by performing a random walk. However in BOA, all solutions (butterflies) are equal and each solution performs the very same task, what other solutions are performing which makes BOA different from ABC conceptually as well as mathematically. Moreover, in BOA, the movement of a butterfly is based on fragrance emitted by other butterflies. The movement is further divided into two phases, one is search for food or mating partner and other is local search. The choice of the phase depends upon a random number which is compared to a fixed value of switch probability. Search for food or mating partner is dominated by the fragrance of other butterflies in the search space which differentiates BOA from ABC algorithm.

There are some other population-based algorithms like DE and CS which share some of the features of BOA inevitably, but they are inspired by completely different disciplines of biology. DE uses a different kind of recombination and decomposition operator to manipulate the solutions. CS is inspired by obligate blood parasitism of cuckoo bird by laying their eggs in the nest of other species bird. In CS, an alien egg/solution is detected by probability and if detected, then that solution is discarded. Another significant difference is that in CS, lèvy flight is used instead of random walk to generate new solutions.

The above comparison between BOA and other metaheuristic algorithms reveal that to the best of our knowledge, there is no metaheuristic algorithm in the literature which mimics the food foraging behavior of the butterflies. This motivated our attempt to propose a new nature-inspired metaheuristic algorithm inspired by the biological behavior of the butterflies with an aim to solve a broader range of optimization problems.

4 Optimization testbed and experimental platforms

Every novel optimization algorithm must be subjected to benchmark optimization functions and its performance must be compared with other existing popular algorithms. There are various benchmark optimization test functions available; however, there is no standardized set of benchmark functions which is agreed upon for validating new algorithms. In order to validate proposed Butterfly Optimization Algorithm (BOA), a diverse subset of such benchmark functions is chosen in this paper as shown in Table 1. To validate and benchmark the performance of BOA, simulations on 30 different benchmark functions are conducted. These testbed benchmark functions are chosen from the benchmark set proposed in (Yao et al. 1999; Liang et al. 2013) to determine various features of the algorithm such as fast convergence, attainment of a large number of local minima points, ability to jump out of local optima and avoid premature convergence.

BOA is implemented in C++ and compiled using Nokia Qt Creator 2.4.1 (MinGW) under Microsoft Windows 8 operating system. All simulations are carried out on a computer with an Intel(R) Core(TM) i5 − 3210 @ 2.50 GHz CPU. In each run, we use a maximum number of 10, 000 iterations as the termination criterion of BOA. In order to reduce statistical errors and generate statistically significant results, each function is repeated for 30 Monte Carlo runs. The mean, standard deviation, best, median, and worst results of BOA are recorded.

In order to meet the requirement set by Liang et al. (2013), we use one fixed combination of parameters for BOA in the simulation of all benchmark functions. The population size n is 50, modular modality c is 0.01 and power exponent a is increased from 0.1 to 0.3 over the course of iterations. To start with, we can use $p = 0.5$ as an initial value and then do a parametric study to find the most appropriate parameter range. From our simulations, we found that $p = 0.8$ works better for most applications.

To evaluate the performance of our proposed butterfly optimization algorithm, it is compared to those algorithms which are the best performing and produce a satisfactory performance when applied to global optimization problems. So considering these factors, seven optimization algorithms have been chosen namely, ABC (Karaboga and Basturk 2007), CS (Yang and Deb 2009), DE (Storn and Price 1997), FA (Yang 2010b), GA (Goldberg and Holland 1988), MBO (Wang et al. 2015) and PSO (Eberhart and Shi 2001). These algorithms are widely employed to compare the performance of optimization algorithms.

All the experiments are conducted under the same conditions in order to obtain fair results as shown in Wang et al. (2014). In this study, ABC employs only one control parameter $limit$ whose value is calculated by limit $= \mathrm{SN} \times D$ where D is the dimension of the problem and SN is the number of food sources or employed bees. For CS, we used a probability $p_a = 0.8$. For DE, we used a weighting factor $F = 0.5$ and a crossover constant $\mathrm{CR} = 0.5$. For FA, $\alpha = 0.2$, $\beta_0 = 1$ and $\gamma = 1$. For GA, we used roulette wheel selection, single point crossover with a crossover probability of 1, and a mutation probability of 0.01.

For MBO, we used max step $S_{\max} = 1.0$, butterfly adjusting rate $\mathrm{BAR} = 5/12$, migration period peri $= 1.2$, and the migration ratio $p = 5/12$. For PSO, we used only global learning (no local neighborhoods), an inertial constant $= 0.3$, a cognitive constant $= 1$, and a social constant for swarm interaction $= 1$. These parameters are set as reported by the authors in the past (Karaboga and Basturk 2007; Yang and Deb 2009; Storn and Price 1997; Kalra and Arora 2016; Huang et al. 2007; Goldberg and Holland 1988; Wang et al. 2015; Eberhart and Shi 2001). The rigorous nonparametric statistical framework is used to compare the performance of BOA with other selected optimization algorithms. For each run of the algorithm, all initial solutions of the population are randomly generated.

5 Simulation results

In this section, simulation results of BOA on benchmark functions selected in Sect. 4 are presented. A comparison is performed among BOA and other metaheuristic algorithms, and statistical analysis on simulation results is given. The detailed simulation results of BOA on benchmark functions selected in Sect. 4 are presented in Table 2. The benchmark functions are selected in such a way that they can assess the algorithm's ability to converge fast, jump out of local optima, ability to achieve a large number of local optima and avoid premature convergence. The mean and standard deviation values obtained by BOA and other algorithms on various testbed benchmark functions are listed in Table 3. The mean values in bold indicate superiority. The simulation results indicate that BOA generally gives very outstanding performance compared with other algorithms. Further pairwise Wilcoxon rank test at 95% confidence interval is used to figure out partial ordering of algorithms for these selected benchmark functions (Maesono 1987). Outcomes of pairwise Wilcoxon rank test statistical comparison for all benchmark functions are shown in Table 4 with the position of BOA in bold. If algorithms Algo-1, Algo-2, and Algo-3 perform better than Algo-4, we conclude Algo-1, Algo-2, and Algo-3 outperform Algo-4 on that particular benchmark function.

The results of partial orderings of the algorithms are constructed based on the pairwise findings are shown in Table 5, where it is seen that BOA shows an outstanding performance in unimodal benchmark functions when compared with other algorithms in terms of both mean value of the results and

the Wilcoxon rank test result. This indicates the outstanding capability of BOA in fast converging to the global optimum while avoiding premature convergence. BOA also shows its dominating performance in most of the multimodal functions and satisfactory results in some functions like f_7, f_9 and f_{10}. Special attention should be paid to the noisy (Quartic) problems as these challenges occur frequently in real-world applications. BOA provided a significant performance boost on these noisy problems.

Besides optimization accuracy, convergence speed is quite essential to an optimizer. To validate the convergence speed of the BOA, we conducted more thorough experiments. Figures 2, 3, 4, 5, 6 and 7 depict the convergence curves of BOA and other algorithms on 100 iterations. From these results, we can arrive at a conclusion that BOA can find excellent solutions with only 100 iterations which reflect the fast convergence speed of the proposed BOA.

In order to have a clearer view of the statistical assessment of the simulation results, we further rank the algorithms based on their performance in the Wilcoxon rank test as shown in Table 4. For each function, the first algorithm is assigned a rank value 1, the second with 2 and so on. For the ties, an average rank is given to the algorithms involved in the tie. Take f_2 as an example, BOA, DE, GA and PSO shares the same rank, and they are the first to the fourth order algorithms in this function. So we assign an average rank value of 2.5 to all these four algorithms. We then sum up the rank values of each algorithm to have an overall assessment of the performance of the algorithms. Similar evaluation methods have been adopted in previous metaheuristic algorithm tests (Shilane et al. 2008). The test results which are presented

Table 2 Simulation results of BOA

Function	Butterfly optimization algorithm				
	Mean	SD	Best	Median	Worst
f_1	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
f_2	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
f_3	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
f_4	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
f_5	3.8917E−05	2.9003E−05	5.8800E−05	3.5850E−05	1.2000E−05
f_6	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
f_7	1.7183E+00	0.0000E+00	1.7183E+00	1.7183E+00	1.7183E+00
f_8	1.8472E−19	2.6886E−20	1.6300E−19	1.6300E−19	2.1700E−19
f_9	4.4108E−01	5.7467E−02	2.8900E−01	3.7800E−01	5.7400E−01
f_{10}	− 5.3382E+00	− 5.6092E+00	− 5.7700E+00	− 3.6200E+00	− 2.2500E+00
f_{11}	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
f_{12}	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
f_{13}	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
f_{14}	2.8837E+01	3.1281E−02	2.8754E+01	2.8754E+01	2.8927E+01
f_{15}	− 1.0000E+00	0.0000E+00	− 1.0000E+00	− 1.0000E+00	− 1.0000E+00
f_{16}	− 1.8673E+02	2.06493E−11	− 1.8673E+02	1.8673E+02	− 1.8673E+02
f_{17}	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
f_{18}	6.9906E−153	1.4788E−152	3.7983E−155	2.3793E−153	9.7687E−152
f_{19}	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	7.2407E−158
f_{20}	− 2.2662E+03	4.5626E+02	− 2.8790E+03	− 2.3458E+03	− 1.8373E+03
f_{21}	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
f_{22}	3.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
f_{23}	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
f_{24}	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
f_{25}	2.8400E−02	1.3179E−02	8.9600E−03	2.6800E−02	5.4300E−02
f_{26}	− 1.0200E+01	0.0000E+00	− 1.0200E+01	− 1.0200E+01	− 1.0200E+01
f_{27}	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
f_{28}	− 2.7500E+07	5.6500E+07	− 9.6100E+07	− 3.1800E+07	7.3000E+07
f_{29}	− 3.7900E−03	0.0000E+00	− 3.7900E−03	− 3.7900E−03	− 3.7900E−03
f_{30}	1.1527E−06	9.4711E−07	1.0700E−08	6.9200E−07	2.6800E−06

Table 4 Pairwise Wilcoxon rank test results

No.	Wilcoxon rank test order
f_1	**BOA** < DE < CS < PSO < FA < GA < MBO < ABC
f_2	**BOA** ≈ DE ≈ GA ≈ PSO < FA < ABC < CS < MBO
f_3	**BOA** < DE < CS < PSO < FA < GA < MBO < ABC
f_4	**BOA** ≈ FA < PSO < GA < DE < ABC < CS < MBO
f_5	**BOA** < DE < GA < MBO < FA < ABC < PSO < CS
f_6	**BOA** ≈ ABC ≈ DE ≈ FA ≈ GA ≈ MBO < CS < PSO
f_7	FA < GA < **BOA** < DE < ABC < MBO < CS < PSO
f_8	**BOA** < DE < FA < GA < ABC < CS < PSO < MBO
f_9	GA < ABC < DE < **BOA** < FA < PSO < CS < MBO
f_{10}	GA < DE < ABC < FA < MBO < CS < PSO < **BOA**
f_{11}	**BOA** < DE < GA < ABC < FA < MBO < CS < PSO
f_{12}	**BOA** < GA < DE < FA < PSO < MBO < ABC < CS
f_{13}	**BOA** ≈ DE ≈ PSO ≈ GA < ABC < MBO < FA < CS
f_{14}	MBO < FA < DE < **BOA** < GA < CS < ABC < PSO
f_{15}	**BOA** ≈ CS ≈ DE ≈ GA ≈ MBO ≈ PSO < ABC < FA
f_{16}	**BOA** ≈ ABC ≈ CS ≈ DE ≈ FA ≈ GA ≈ PSO < MBO
f_{17}	**BOA** < DE < GA < ABC < CS < FA < PSO < MBO
f_{18}	**BOA** < FA < CS < GA < ABC < PSO < DE < MBO
f_{19}	**BOA** < DE < FA < MBO < GA < ABC < CS < PSO
f_{20}	DE < MBO < GA < ABC < PSO < FA < **BOA** < CS
f_{21}	**BOA** ≈ DE ≈ FA ≈ GA ≈ PSO < ABC < CS < MBO
f_{22}	**BOA** ≈ ABC ≈ CS ≈ DE ≈ FA ≈ GA ≈ MBO ≈ PSO
f_{23}	**BOA** ≈ DE ≈ FA ≈ MBO ≈ PSO < GA < CS < ABC
f_{24}	**BOA** < FA < GA < CS < ABC < DE < PSO < MBO
f_{25}	FA < GA < PSO < DE < CS < **BOA** < ABC < MBO
f_{26}	**BOA** ≈ ABC ≈ DE ≈ FA ≈ PSO < GA < CS < MBO
f_{27}	**BOA** < DE < FA< GA < ABC < CS < PSO < MBO
f_{28}	**BOA** < FA < DE < GA < CS < MBO < ABC < PSO
f_{29}	**BOA** ≈ ABC ≈ DE ≈ FA ≈ GA ≈ PSO ≈ CS < MBO
f_{30}	PSO < **BOA** < GA < FA < CS < DE < ABC < MBO

Table 5 Rank summary of statistical assessment results

No.	BOA	ABC	CS	DE	FA	GA	MBO	PSO
f_1	1	8	3	2	5	6	7	4
f_2	2.5	6	7	2.5	5	2.5	8	2.5
f_3	1	8	3	2	5	6	7	4
f_4	1.5	6	7	5	1.5	4	8	3
f_5	1	6	8	2	5	3	4	7
f_6	3.5	3.5	7	3.5	3.5	3.5	3.5	8
f_7	3.5	5	7	3.5	1	2	6	8
f_8	1	5	6	2	3	4	8	7
f_9	4	2	7	3	5	1	8	6
f_{10}	8	3	6	2	4	1	5	7
f_{11}	1	4	7	2	5	3	6	8
f_{12}	1	7	8	3	4	2	6	5
f_{13}	2	5	8	2	7	4	6	2
f_{14}	4	7	6	3	2	5	1	8
f_{15}	3.5	7	3.5	3.5	8	3.5	3.5	3.5
f_{16}	4	4	4	4	4	4	8	4
f_{17}	1	4	5	2	6	3	8	7
f_{18}	1	5	3	7	2	4	8	6
f_{19}	1	6	7	2	3	5	4	8
f_{20}	7	4	8	1	6	3	2	5
f_{21}	3	6	7	3	3	3	8	3
f_{22}	4.5	4.5	4.5	4.5	4.5	4.5	4.5	4.5
f_{23}	3	8	7	3	3	6	3	3
f_{24}	1	5	4	6	2	3	8	7
f_{25}	6	7	5	4	1	2	8	3
f_{26}	3	3	7	3	3	6	8	3
f_{27}	1	5	6	2	3	4	8	7
f_{28}	1	7	5	3	2	4	6	8
f_{29}	3.5	3.5	7	3.5	3.5	3.5	8	3.5
f_{30}	2	7	5	6	4	3	8	1
Total	80.5	161.5	178	95	114	108.5	186.5	156

in Tables 4 and 5 depict that BOA is the most effective at finding function minima by performing best on 23 out of a fairly large subset of 30 benchmark functions. DE is the second most effective, followed by GA, FA, PSO, ABC, CS and MBO respectively.

The rank summary of the statistical assessment result supports our previous observations. BOA is the best performing algorithm. It possesses a dominating position in the overall comparison as well. From Table 5, it is observed that no other algorithm can have as stable a performance as BOA. The performance of DE, GA, and FA are satisfactory but neither of them can catch up with BOA in unimodal and multimodal functions ranks. So we conclude that in general, BOA has both best optimization performance and highest stability.

These simulation results discussed in this paper should not be taken to mean that BOA is "better" than other metaheuristic algorithms. Such a general statement would be an oversimplification, especially in view of the no free lunch theorem (Wolpert and Macready 1997). However, the results presented here show that BOA provides better performance than most of the other popular optimization algorithms for the particular benchmarks considered in this paper. The results shown here indicate that BOA is competitive with other metaheuristic algorithms, and could be used to solve various real life problems.

6 BOA for classical engineering problems

This section discusses three engineering design problems, i.e., spring design, welded beam, and gear train design which are employed to evaluate the capability of BOA. Constraint handling is a challenging task for an algorithm when the fitness function directly affects the position updation of the search agents. These classical engineering problems have several equality and inequality constraints which evaluate the capability of BOA from the perspective of constraint handling in order to optimize constrained problems as well.

6.1 Spring design

The objective of this problem is to minimize the weight of a spring as illustrated in Fig. 8 (Arora 2004; Belegundu and Arora 1985; Coello and Montes 2002). The minimization process is subject to some constraints such as shear stress, surge frequency, and minimum deflection. There are three variables in this problem: wire diameter (d), mean coil diameter (D), and the number of active coils (N). The mathematical formulation of this problem is as follows:

$$
\begin{aligned}
&\text{Consider } \vec{x} = [x_1 x_2 x_3] = [dDN],\\
&\text{Minimize } f(\vec{x}) = (x_3 + 2)x_2 x_1^2,\\
&\text{Subject to } g_1(\vec{x}) = 1 - \frac{x_2^3 x_3}{71785 x_1^4} \leq 0,\\
&g_2(\vec{x}) = \frac{4x_2^2 - x_1 x_2}{12,566(x_2 x_1^3 - x_1^4)} + \frac{1}{5108 x_1^2} \leq 0,\\
&g_3(\vec{x}) = 1 - \frac{140.45 x_1}{x_2^2 x_3} \leq 0,\\
&g_4(\vec{x}) = \frac{x_1 + x_2}{1.5} - 1 \leq 0,\\
&\text{Variable range } 0.05 \leq x_1 \leq 2.00,\\
&0.25 \leq x_2 \leq 1.30,\\
&2.00 \leq x_3 \leq 15.00
\end{aligned}
\tag{4}
$$

This problem has been tackled by both mathematical and heuristic approaches. The simulation results of BOA are compared with Grey Wolf Optimization (GWO) (Mirjalili et al. 2014), Gravitational Search Algorithm (GSA) (Rashedi et al. 2009), PSO (He and Wang 2007), ES (Mezura-Montes and Coello 2008), GA (Coello 2000a), HS (Mahdavi et al. 2007), and DE (Huang et al. 2007). The mathematical approaches that have been used to tackle this problem are numerical optimization technique, i.e., mathematical optimization technique (Belegundu and Arora 1985) and constraints correction at constant cost (Arora 2004). The bold values indicate the best one among all methods. Table 6 suggests that BOA finds a design with the minimum weight for this problem.

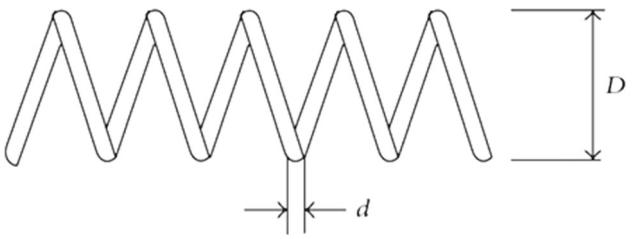

Fig. 8 Schematic representation of spring

6.2 Welded beam design

The objective of this problem is to minimize the fabrication cost of a welded beam as shown in Fig. 9 (Coello 2000a). The constraints are as follows:

1. Shear stress (s)
2. Bending stress in the beam (h)
3. Buckling load on the bar (P_c)
4. End deflection of the beam (d)
5. Side constraints

This problem has four variables such as the thickness of weld (h), the length of attached part of the bar (l), the height of the bar (t), and thickness of the bar (b). The mathematical formulation is as follows:

$$
\begin{aligned}
&\text{Consider } \vec{x} = [x_1 x_2 x_3 x_4] = [hltb],\\
&\text{Minimize } f(\vec{x}) = 1.10471 x_1^2 x_2 + 0.04811 x_3 x_4 (14.0 + x_2),\\
&\text{Subject to } g_1(\vec{x}) = \tau(\vec{x}) - \tau_{\max} \leq 0,\\
&g_2(\vec{x}) = \sigma(\vec{x}) - \sigma_{\max} \leq 0,\\
&g_3(\vec{x}) = \delta(\vec{x}) - \delta_{\max} \leq 0,\\
&g_4(\vec{x}) = x_1 - x_4 \leq 0,\\
&g_5(\vec{x}) = P - P_c(\vec{x}) \leq 0,\\
&g_6(\vec{x}) = 0.125 - x_1 \leq 0,\\
&g_7(\vec{x}) = 1.10471 x_1^2 + 0.04811 x_3 x_4 (14.0 + x_2) - 5.0 \leq 0,\\
&\text{Variable range } 0.1 \leq x_1 \leq 2,\\
&0.1 \leq x_2 \leq 10,\\
&0.1 \leq x_3 \leq 10,\\
&0.1 \leq x_4 \leq 2
\end{aligned}
\tag{5}
$$

$$
\text{where } \tau(\vec{x}) = \sqrt{(\tau')^2 + 2\tau'(\tau'')\frac{x_2}{2R} + \tau''^2}
$$

$$
\tau' = \frac{P}{\sqrt{2} x_1 x_2}
$$

$$
\tau'' = \frac{MR}{J}
$$

Table 6 Comparison of results for spring design problem

Algorithm	Optimum variables			Optimum weight
	d	*D*	*N*	
BOA	**0.051343**	**0.334871**	**12.922700**	**0.0119656**
GWO	0.051690	0.356760	11.288110	0.0126615
GSA	0.050276	0.323680	13.525410	0.0127022
PSO	0.051728	0.357644	11.244543	0.0126747
ES	0.051989	0.363965	10.890522	0.0126810
GA	0.051480	0.351661	11.632201	0.0127048
HS	0.051154	0.349871	12.076432	0.0126706
DE	0.051609	0.354714	11.410831	0.0126702
Mathematical optimization	0.053396	0.399180	9.1854000	0.0127303
Constraint correction	0.050000	0.315900	14.250000	0.0128334

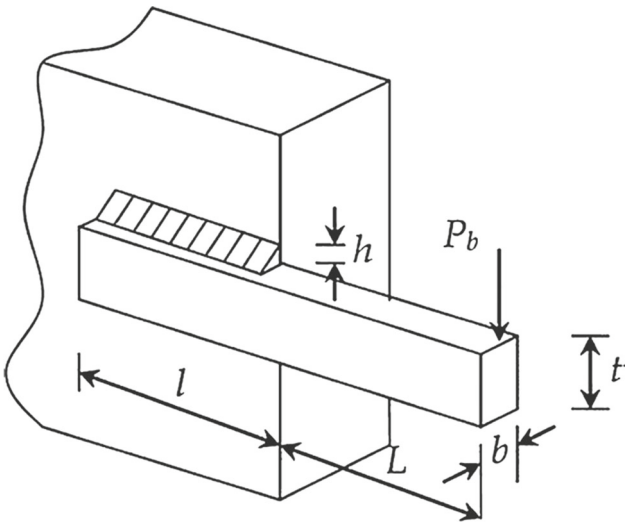

Fig. 9 Schematic representation of welded beam

$$M = P\left(L + \frac{x_2}{2}\right)$$

$$R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2}$$

$$J = 2\left\{\sqrt{2}x_1x_2\left[\frac{x_2^2}{4}\left(\frac{x_1 + x_3}{2}\right)^2\right]\right\}$$

$$\sigma(\vec{x}) = \frac{6\mathrm{PL}}{x_4x_3^2}$$

$$\delta(\vec{x}) = \frac{4\mathrm{PL}^3}{\mathrm{E}x_3^2x + x_4}$$

$$P_c(\vec{x}) = \frac{4.013\mathrm{E}\sqrt{\frac{x_3^2x_4^6}{36}}}{\mathrm{L}^2}\left(1 - \frac{x_3}{2\mathrm{L}}\sqrt{\frac{\mathrm{E}}{4G}}\right)$$

$\mathrm{P} = 6000$ lb, $\mathrm{L} = 14$ in., $\delta_{\max} = 0.25$ in.,

$\mathrm{E} = 30 \times 10^6$ psi,

$$\mathrm{G} = 12 \times 10^6 \text{ psi}, \tau_{\max} = 13600 \text{ psi},$$
$$\sigma_{\max} = 30000 \text{ psi} \tag{6}$$

In the past, this problem is solved by GWO (Mirjalili et al. 2014), GSA (Rashedi et al. 2009), GA1 (Coello 2000b), GA2 (Deb 1991), GA3 (Deb 2000) and HS (Lee and Geem 2005). Richardson's random method, Simplex method, Griffith and linear approximation (Approx) and David's method are the traditional methods that have been employed by Ragsdell and Philips to solve this problem (Ragsdell and Phillips 1976). The comparison results are provided in Table 7 where bold values indicate superiority. The results show that BOA finds a design with the minimum cost compared to others.

6.3 Gear train design

Gear train design problem is a discrete optimization problem. It was introduced by Sandgren (1990), and it has four integer variables between 12 and 60. This problem consists of cost minimization of the gear ratio of a compound gear train (see Fig. 10). The gear ratio can be defined as:

$$\text{Gear ratio} = \frac{\text{Angular velocity of the output shaft}}{\text{Angular velocity of the input shaft}} \tag{7}$$

where T_i denotes the number of teeth of the ith gear wheel which should be integers. It is required to obtain the teeth numbers on the wheels so that a gear ratio reaches to $1/6.931$. Therefore, the mathematical formulation of the objective function is as follows:

$$f(T_a, T_b, T_d, T_f) = \left(\frac{1}{6.931} - \frac{T_bT_d}{T_aT_f}\right)^2 \tag{8}$$

This problem has also been popular among researchers and optimized in various studies. The heuristic methods that have been adopted to optimize this problem are: Chaotic variant of Accelerated PSO (CAPSO) (Gandomi et al. 2013b),

Table 7 Comparison results of welded beam design problem

Algorithm	Optimum variables				Optimum cost
	h	*l*	*t*	*b*	
BOA	**0.1736**	**2.9690**	**8.7637**	**0.2188**	**1.6644**
GWO	0.2056	3.4783	9.0368	0.2057	1.7262
GSA	0.1821	3.8569	10.0000	0.2023	1.8799
GA1	N/A	N/A	N/A	N/A	1.8245
GA2	N/A	N/A	N/A	N/A	2.3800
GA3	0.2489	6.1730	8.1789	0.2533	2.4331
HS	0.2442	6.2231	8.2915	0.2443	2.3807
Random	0.4575	4.7313	5.0853	0.6600	4.1185
Simplex	0.2792	5.6256	7.7512	0.2796	2.5307
David	0.2434	6.2552	8.2915	0.2444	2.3841
Approx	0.2444	6.2189	8.2915	0.2444	2.3815

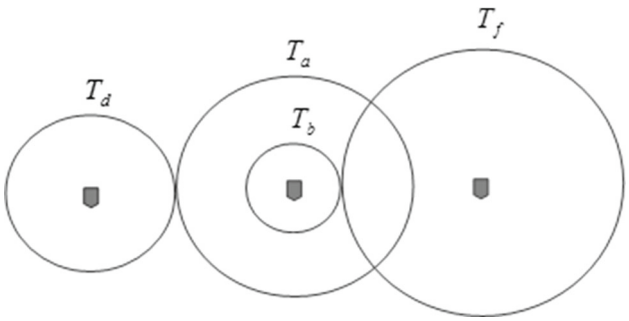

Fig. 10 Schematic representation of gear

CS (Gandomi et al. 2013a), PSO (Parsopoulos and Vrahatis 2005), Genetic Adaptive Search (GeneAS) (Deb and Goyal 1996) and Simulated annealing (Zhang and Wang 1993). Various other approaches used to solve to this problem are nonlinear integer and discrete programming (Sandgren 1990), sequential linear approach (Loh and Papalambros 1991), mixed integer discrete continuous optimization (Kannan and Kramer 1994), discrete continuous programming approach (Fu et al. 1991) and evolutionary programming (Cao and Wu 1997). Table 8 demonstrates the results for the best objective value obtained by above-mentioned algorithms in this study. As it can be seen from the Table 8, the best result obtained by the proposed BOA in this study is equivalent to the results in the literature.

7 Conclusion

In this paper, a novel Butterfly Optimization Algorithm is proposed to solve global optimization problems. BOA is based on food foraging behavior and information sharing strategy of butterflies. BOA is conceptually very simple and

Table 8 Comparison of results for gear train design problem

Algorithm	Optimum variables				Gear ratio	$f_{\min}$
	T_a	T_b	T_d	T_f		
BOA	**43**	**16**	**19**	**49**	**0.1442**	2.701×10^{-12}
CAPSO	49	19	16	43	0.1442	2.701×10^{-12}
CS	43	16	19	49	0.1442	2.701×10^{-12}
PSO	43	16	19	49	0.1442	2.701×10^{-12}
Sequential linearization approach	42	16	19	50	0.1447	0.23×10^{-6}
GeneAS	33	14	17	50	0.1442	1.362×10^{-9}
Mixed-variable evolutionary programming	52	15	30	60	0.1442	2.36×10^{-9}
Simulated annealing	52	15	30	60	0.1442	2.36×10^{-9}
Mixed integer discrete continuous programming	47	29	14	59	0.1464	4.5×10^{-6}
GA	N/A	N/A	N/A	N/A	N/A	2.33×10^{-7}
Mixed integer discrete continuous optimization	33	15	13	41	0.1441	2.146×10^{-8}
Nonlinear integer and discrete programming	45	22	18	60	0.1467	5.712×10^{-6}

easy to implement. The performance of the proposed BOA is compared with well-known optimization algorithms such as ABC, CS, DE, FA, GA, MBO, and PSO, by experimenting with 30 benchmark problems with different characteristics like multimodality, separability, regularity, and dimensionality. The usefulness of BOA is also evaluated by solving three engineering design problems (spring design, welded beam design, and gear train design) which have different natures of objective functions, constraints and decision variables. Simulation results indicate that the performance of the BOA is promising since it has produced competitive results in comparison with the other studied algorithms and it has an ability to become an effective tool for solving real word optimization problems.

Future research on BOA can be divided into two parts: research on algorithm and research on real-world application. In terms of algorithm research, BOA has the potential to solve combinatorial problems. Many metaheuristic algorithms such as GA, PSO were originally designed to solve continuous optimization problems and later modified to solve the combinatorial problem. So it would be an interesting area to modify BOA and analyze results when applied to combinatorial problems. Binary and multi-objective versions of this algorithm may be developed to solve discrete and multi-objective problems. Even though BOA has few parameters, i.e., sensor modality, power exponent and switch probability, but still, it is very interesting to develop self-adaptive schemes for BOA in order to reduce the effort in tuning parameters. Another interesting dimension can be identifying real-world applications which can be efficiently solved using BOA.

Acknowledgements The authors acknowledge the contribution of I. K. Gujral Punjab Technical University, Kapurthala, Punjab, India.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

References

Arora JS (2004) Introduction to optimum design. Elsevier Academic Press, England

Arora S, Singh S (2013a) The firefly optimization algorithm: convergence analysis and parameter selection. Int J Comput Appl 69(3):48–52

Arora S, Singh S (2013b) A conceptual comparison of firefly algorithm, bat algorithm and cuckoo search. In: 2013 international conference on control computing communication and materials (ICCCCM), IEEE, pp 1–4

Arora S, Singh S, Singh S, Sharma B (2014) Mutated firefly algorithm. In: 2014 international conference on parallel, distributed and grid computing (PDGC), IEEE, pp 33–38

Back T (1996) Evolutionary algorithms in theory and practice. Oxford University Press, Oxford

Baird JC, Noma EJ (1978) Fundamentals of scaling and psychophysics. Wiley, Hoboken

Belegundu AD, Arora JS (1985) A study of mathematical programming methods for structural optimization. Part I: theory. Int J Numer Methods Eng 21(9):1583–1599

Blair RB, Launer AE (1997) Butterfly diversity and human land use: species assemblages along an urban grandient. Biol Conserv 80(1):113–125

Brownlee J (2011) Clever algorithms: nature-inspired programming recipes, 1st edn. LuLu. ISBN 978-1-4467-8506-5

Cao Y, Wu Q (1997) Mechanical design optimization by mixed-variable evolutionary programming. In: IEEE conference on evolutionary computation, IEEE Press, p 443–446

Coello CAC (2000a) Use of a self-adaptive penalty approach for engineering optimization problems. Comput Ind 41(2):113–127

Coello CA (2000b) Constraint-handling using an evolutionary multi-objective optimization technique. Civil Eng Syst 17(4):319–346

Coello CAC, Montes EM (2002) Constraint-handling in genetic algorithms through the use of dominance-based tournament selection. Adv Eng Inform 16(3):193–203

Deb K (1991) Optimal design of a welded beam via genetic algorithms. AIAA J 29(11):2013–2015

Deb K (2000) An efficient constraint handling method for genetic algorithms. Comput Methods Appl Mech Eng 186(2):311–338

Deb K, Goyal M (1996) A combined genetic adaptive search (GeneAS) for engineering design. Comput Sci Inf 26:30–45

Eberhart RC, Shi Y (2001) Particle swarm optimization: developments, applications and resources. In: Proceedings of the 2001 congress on evolutionary computation, 2001, vol 1. IEEE, pp 81–86

Fister I Jr, Yang X-S, Fister I, Brest J, Fister D (2013) A brief review of nature-inspired algorithms for optimization. arXiv preprint arXiv:1307.4186

Fu J-F, Fenton RG, Cleghorn WL (1991) A mixed integer-discrete-continuous programming method and its application to engineering design optimization. Eng Optim 17(4):263–280

Gandomi AH, Yang X-S, Alavi AH (2013a) Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems. Eng Comput 29(1):17–35

Gandomi AH, Yun GJ, Yang X-S, Talatahari S (2013) Chaos-enhanced accelerated particle swarm optimization. Commun Nonlinear Sci Numer Simul 18(2):327–340

Gazi V, Passino KM (2004) Stability analysis of social foraging swarms. Syst Man Cybern Part B: Cybern IEEE Trans 34(1):539–557

Goldberg DE, Holland JH (1988) Genetic algorithms and machine learning. Mach Learn 3(2):95–99

Gupta S, Arora S (2015) A hybrid firefly algorithm and social spider algorithm for multimodal function. Intell Syst Technol Appl 1:17

He Q, Wang L (2007) An effective co-evolutionary particle swarm optimization for constrained engineering design problems. Eng Appl Artif Intell 20(1):89–99

Holland JH (1992) Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence. MIT press, Cambridge

Huang F-Z, Wang L, He Q (2007) An effective co-evolutionary differential evolution for constrained optimization. Appl Math Comput 186(1):340–356

Kalra S, Arora S (2016) Firefly algorithm hybridized with flower pollination algorithm for multimodal functions. In: Proceedings of the international congress on information and communication technology, Springer, Singapore, pp 207–219

Kannan B, Kramer SN (1994) An augmented lagrange multiplier based method for mixed integer discrete continuous optimization and its applications to mechanical design. J Mech Des 116(2):405–411